



**Váci SZC Petőfi Sándor Műszaki Technikum,  
Gimnázium és Kollégium**



**SZOFTVERFEJLESZTŐ ÉS TESZTELŐ**

**Záródolgozat**

***Cím:*** MInerva – „Bátor? Okos? Bölcs?”

***Készítette:*** Berdó Tamás

***Csapattársak:*** Füleki József, Sartner Bettina

***Konzulens:*** Molnár Máté Norbert

**2025**

## Nyilatkozat

Alulírott, . . . . . tanuló kijelentem, hogy ez a szakdolgozat saját tudásom, önálló munkám eredménye. A felhasznált eszközöket és szakirodalmakat (tanulmány, internetes forrás, személyes közlés stb.) idézőjel és megjelölt források nélkül nem építettem be, egyértelműen megjelöltem. Az elkészült záródolgozatban található eredményeket a képzőintézmény és a feladatot kiíró intézmény saját céljára térítés nélkül felhasználhatja, a titkosításra vonatkozó esetleges megkötések mellett.

Aszód, 2025 . . . . .

Váci SZC PETŐFI SÁNDOR Műszaki Technikum, Gimnázium és Kollégium

**KONZULTÁCIÓS LAP****Szoftverfejlesztő és tesztelő**A tanuló neve: Berdó Tamás Osztálya: 13. D.A záródolgozat témája: Minerva – „Eltér? Olyan? Pólya?”A Minerva egy interaktív mesterséges intelligencia alapú tudáspolitforma és közösségi tér tanulók és tanárok közötti számára.

Sorszám:	A konzultáció időpontja:	A konzulens aláírása:
1.	2024.11.23	Molnár Máté
2.	2025.01.31	Molnár Máté
3.	2025.02.25	Molnár Máté
4.		
5.		

A záródolgozat beadható.

Aszód, .....

Molnár Máté  
.....  
konzulens/

# Tartalom

SZOFTVERFEJLESZTŐ ÉS TESZTELŐ .....	2
Záródolgozat .....	2
Nyilatkozat.....	3
Bevezető .....	7
1. Felhasználói dokumentáció .....	8
1.1. A program általános specifikációja .....	8
1.2. A program rendszerkövetelményei .....	9
1.2.1. Hardver követelmények .....	9
1.2.2. Szoftver követelmények.....	10
1.3. A program elérése, „telepítése” .....	11
1.4. A program használatának részletes leírása.....	12
1.4.1. Főoldal.....	12
1.4.2. Rólunk.....	15
1.4.3. Regisztráció.....	16
1.4.4. Bejelentkezés.....	19
1.4.5. Beszéljess! .....	22
1.4.6. Fórum .....	25
1.4.7. Fiókom .....	27
1.4.8. Az oldal nem található.....	29
2. Fejlesztői dokumentáció.....	30
2.1. Témaválasztás .....	30
2.2. Alkalmazott fejlesztői eszközök.....	31
2.2.1. Programok, szoftverek, alkalmazások:.....	31
2.2.2. Backend Modulok: .....	34
2.2.3. Frontend modulok: .....	35
2.2.4. Programozási nyelvek .....	37
2.2.5. Leírónyelvek, nyelvi kiterjesztések.....	37
2.3. Adatmodell, Adatbázis felépítés .....	38
2.3.1. „credentials” tábla, és felépítése: .....	38
2.3.2. „profileDetails” tábla felépítése: .....	39
2.3.3. „forumMessages” tábla felépítése:.....	40

2.3.4.	„sqlite_sequence” tábla: .....	40
2.3.5.	Adatbázis táblák között lévő kapcsolat: .....	41
2.4.	Részletes feladat-specifikáció, algoritmusok, forráskódok .....	42
2.4.1.	Titkosító funkció: .....	42
2.4.2.	Email küldő funkció: .....	44
2.4.3.	Model tuning betöltő modul: .....	45
2.4.4.	Végpontokat betöltő router.....	47
2.4.5.	Hibátlan indulás.....	49
2.4.6.	Authentikációs folyamat: .....	51
2.5.	Tesztelési dokumentáció .....	52
2.5.1.	Védett hálózati végpont elérésének tesztelése: .....	52
2.5.5.	Egyéb információ: .....	53
2.6.	Továbbfejlesztési lehetőségek.....	55
2.6.1.	Beszédalapú interakció.....	55
2.6.2.	Pontgyűjtési rendszer: .....	55
2.6.3.	Mobilalkalmazás: .....	55
2.6.4.	Személyre szabott ajánlások: .....	55
2.6.5.	Mesterséges intelligencia fejlesztése:.....	55
2.6.6.	Jobb adatbiztonság: .....	56
2.6.7.	Partnerségek iskolákkal és egyetemekkel: .....	56
2.6.8.	Küzdelem a magány ellen .....	56
3.	Összegzés .....	58
4.	Forrásmegjelölés .....	60
4.1.	Dokumentációk: .....	60
4.2.	Egyéb források: .....	60
5.	Ábrajegyzék .....	61

## Bevezető

A szakdolgozat témáját a tanulási nehézségek és a mai fiatalokat egyre gyakrabban érintő figyelemzavarok inspirálták, amelyeket főként a közösségi platformok, mint a TikTok, YouTube Shorts és Instagram használata erősítenek.

A töménytelen mennyiségű gyors és rövid videós tartalmak fogyasztása miatt megváltozott a fiatalok koncentrációs képessége, egyre nehezebbé válik a tartós figyelem fenntartása és a mélyebb tanulási folyamatokba való belemerülés. Ezt a változást magamon először az érettségi időszakra készülődve vettem észre, mikor egy hosszabb Wikipédia oldal böngészése közben előfordult, hogy tízszer is elterelődött a figyelmem.

A projekt, amelyet MInerva – „Bátor? Okos? Bölcs?” néven indítottunk el, kifejezetten erre a problémára próbál megoldást kínálni. Az oldal célja, hogy modern, mesterséges intelligenciára épülő megoldásokkal segítsen a diákoknak, tanároknak és szülőknek a tanulásban, és támogassa a koncentráció és a tananyag mélyebb megértésének fejlesztését. Ezt úgy próbáljuk meg elérni, hogy az MI segítségével életre keltettük híres történelmi személyeinket, például Arany Jánost vagy Petőfi Sándort. A felhasználók úgy beszélgethetnek velük, mintha egy modern chatalkalmazásban csevegnének, közben számos érdekes és fontos információt sajátíthatnak el. Ez a megközelítés nemcsak informatív, hanem szórakoztató és leköti a tanuló figyelmét, miközben a költő korszakában érezheti magát.

A közeljövőben nem csak írásban, szóban is társaloghatunk majd a kedvenc személyeinkkel, így akár játszhatunk vagy beszámolhatunk neki egy napunkról és ő is elmeséli a napját a sajátos, régies beszédstílusával, így tágítva a szókincsünket és memóriánkat.

# **1. Felhasználói dokumentáció**

## **1.1. A program általános specifikációja**

A weboldal egy egyedi megoldást nyújt a figyelem problémákkal rendelkező tanulók oktatásának segítésére. Az oldal mesterséges intelligenciát használva életre kelt történelmi személyeket, költőket, kikkel úgy cseveghetnek a tanulni vágyók egy chates környezetben, mint ha csak egy barátjukkal beszélgetnének.

A mesterséges intelligencia modellünket ellenőrzött adatokkal tanítottuk, ellentétben más alkalmazásokkal vagy internetes chatekkel, melyek a világhálón található összes, akár hamis információkat tartalmazó oldalakkal is tréningelve lettek.

Karaktereinknek egyszerűen tehetnek fel kérdéseket, melyre az a legjobb tudása szerint válaszol, az adott kornak és helyzetnek megfelelően. A kérdéseket természetesen az adott témában kompetens, megfelelő modellnek, vagyis személynek kell feltenni.

Hogy ne legyen unalmas a tanulás, a gyerekek haladhatnak előre megírt történetek alapján, ami a tanmenetnek megfelelő sorrendben következik, vagy akár csoportos környezetben maga a tanár is állíthat össze előrehaladási sorrendet. Ekkor a diákok úgy követhetnek végig egy történelmi eseményt annak minden részletével, mint ha az abban a pillanatban játszódna, és beszélhetik azt meg az adott karakterrel.



## 1.2. A program rendszerkövetelményei

### 1.2.1. Hardver követelmények

*Technikailag bármely eszköz, ami képes egy modern böngésző futtatására, és aktív internet kapcsolattal rendelkezik.*

*A felhasználói élmény maximalizálása érdekében, állítottunk össze egy táblázatot a tökéletes eszköz kiválasztásához.*

	Minimális	Ajánlott
Memória	2GB	4GB
Videokártya	1GB -	2GB graphics acceleration
Processzor	2mag, 2Ghz	4mag, 3Ghz
Kijelző felbontás	1280x720	1920x1080
Érintő képernyő	270x480	360x740
Internetkapcsolat	stabil 3mbps	stabil 10mbps
Mobil hálózat	4G	4G+ vagy 5G

### 1.2.2. Szoftver követelmények

Bármely operációs rendszer, amely képes futtatni a felsorolt böngészők egyikét.

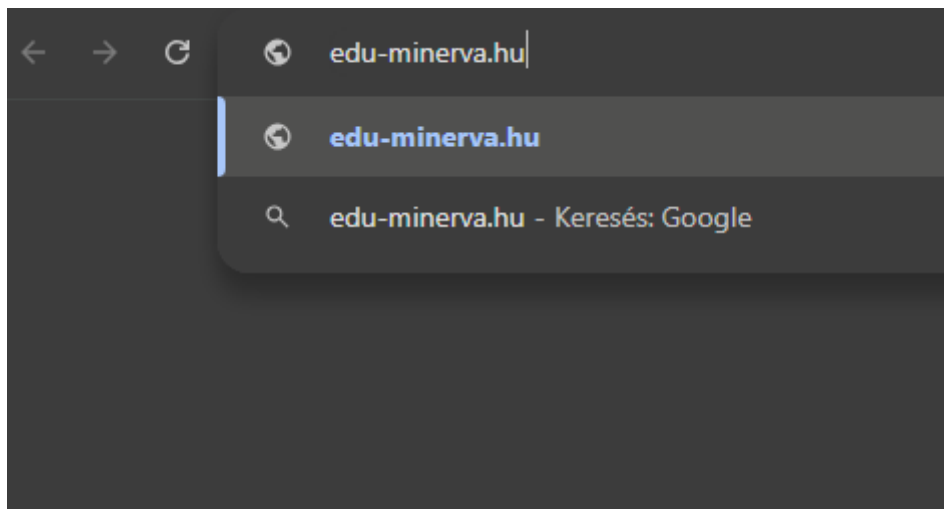
Webböngésző	Minimális verzió
Chrome x86 (Chromium, Brave)	37
Edge	79
Firefox	34
Opera	24
Safari	7
Chrome arm (Chrome Mobile, Brave)	37
Firefox arm	34
Opera arm	24
Safari iOS	7
Samsung Internet	3
WebView Android	37
WebView iOS	7

*Biztonsági és kompatibilitási okból ajánlott mindig a legújabb stabil kiadást használni.*

### 1.3. A program elérése, „telepítése”

Oldalunk egy magyar webcímen került publikálásra, így internetkapcsolat mellett bármikor könnyedén használhatja mobilról és asztali eszközről egyaránt.

A webalkalmazás eléréséhez nyissa meg a fentebb említett böngészők megfelelő verziójának egyikét, és gépelje a keresősávba a következő webcímet: [www.edu-minerva.hu](http://www.edu-minerva.hu) vagy [edu-minerva.hu](http://edu-minerva.hu).



1. ábra: Az oldal elérési címe

A böngészőn kívül más segédprogramra nincs is szükség az oldal felhasználó szintű használatához, ugyanis a teljes kódrendszer a felhőben került tárolásra és futtatásra.

## 1.4. A program használatának részletes leírása

### 1.4.1. Főoldal

Az oldal betöltését követően a kezdőoldalon találja magát, ennek tetején helyezkedik el a navigációs sáv, amely minden oldalon egyaránt megtalálható.



2. ábra: Főoldal, körhinta elem

Az oldal tetején lévő menüsáv bal oldalán található a további oldalakat, „Regisztráció, Bejelentkezés, Rólunk”, ezekre kattintva át navigálhat az adott oldalra, a jobb oldalon pedig a sötét mód kapcsológombja helyezkedik el. Erre rányomva átállíthatja az oldal témáját éjszakai módra, azaz a weboldal háttérszíne sötétre, a betűk pedig világosra változnak, ezzel megkönnyítve az olvasást az esti órákban, és kíméli a szemét az intenzív világos színektől. A megadott beállítást elmenti a böngésző, így a későbbiekben automatikusan a kiválasztott témát fogja használni az alkalmazás. A navigációs sávon megjelenő ikon annak függvényében változik, hogy éppen melyik téma beállítást alkalmazta. Ez a funkció kliensenként külön állítható, így például használhatja az asztali számítógépét világos, mobil készülékét pedig sötét módban, így OLED alapú kijelzővel felszerelt készülék esetén akkumulátor kapacitást és üzemidőt is spórolhat.



3. ábra: Főoldal mobil nézetben, sötétmódban

Lentebb görgetve a körhinta elem alatt, amit lapozhat a két oldalán található nyíllal, (2. ábra) szövegbuborékokban érdekességeket olvashat Minerváról, az istenségről, majd a buborékok alatt lévő „Kattints ide!” gombra kattintva lenyílik egy kis szöveghasáb még több tudnivalóval, illetve egy kis videós ismertető is megtekinthető, amivel még közelebb kerülhetünk a projekt hangulatához és Minervához. Az oldal automatikusan igazodik, skálázódik a készülék méreteihez, bármely eszközön is nyitja azt meg, kényelemeesen fogja tudni használni.

Mobil nézetben a menüpontok az oldal jobb felső sarkában lévő szendvics elemmel nyithatóak meg.

Az oldal alja felé görgetve találkozunk a „Miért válaszd a MInervát?” alcímmel, ahol már nem az istenségről, hanem a MInerva projektről olvashat érdekességeket, információkat és a főbb célkitűzéseket.

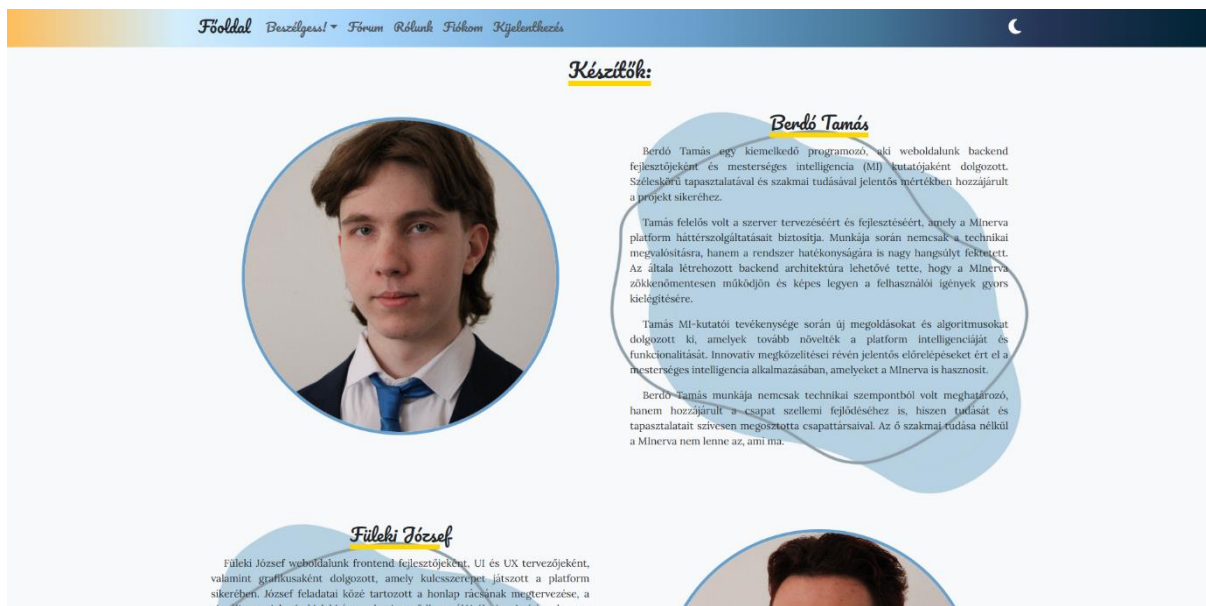
4. ábra: Célkitűzések

Majd az oldal lábán olvashatja az elérhetőséginket, és támogathatja munkásságunk. Ez a modul minden oldal alján ugyan ott helyezkedik el, így bármikor könnyedén megtalálhatja a fontosabb információkat. Elérhetőségeink között megtalálható Facebook, illetve Instagram oldal, ahol a Minerva „brand” népszerűsítésére és divatossá tételére törekszünk, valamint a Discord szerverünk, ahol közvetlen is felveheti velünk a kapcsolatot, és egyedi meghívólinkünk segítségével (<https://dc.edu-minerva.hu/>) könnyedén megoszthatja azt másokkal is! Azonban, ha hivatalos formában szeretne minket megkeresni, azt megteheti a [support@edu-minerva.hu](mailto:support@edu-minerva.hu) e-mail címen, amit szintén az oldal alján megtalál.

5. ábra: Az oldal lábjegyzete

Az adatvédelmi tájékoztatónk is itt olvashatja, ahol minden részletes információt megtalál adatai kezeléséről, jogairól, és az oldallal kapcsolatos szabályokról.

## 1.4.2. Rólunk



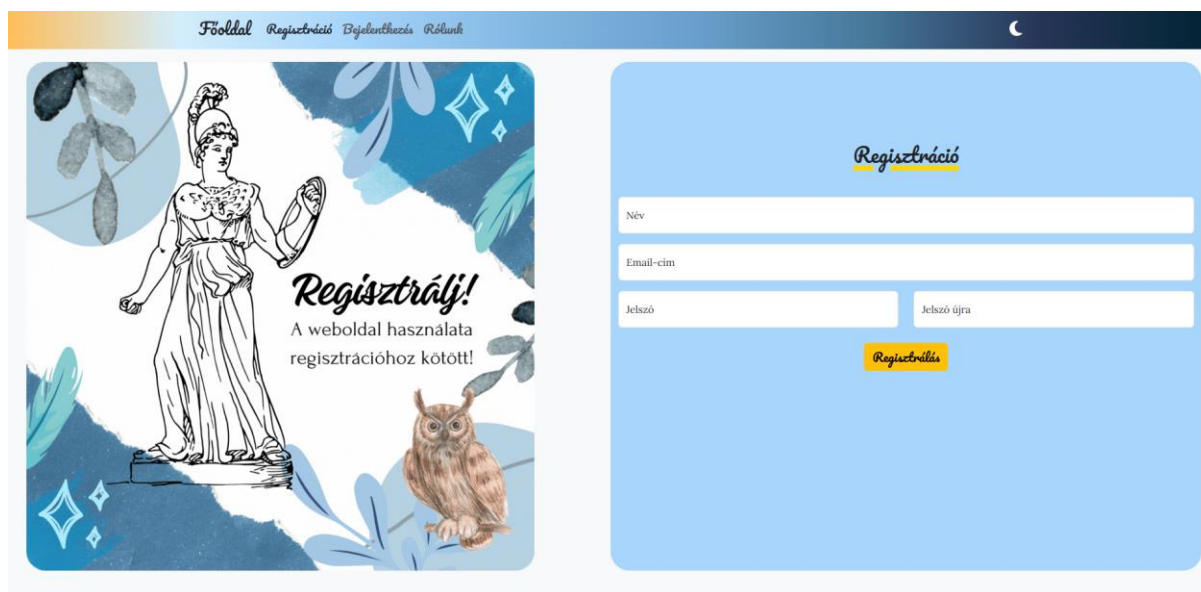
6. ábra: Rólunk oldal

Ezen az oldalon megismerkedhet a készítőkkal és az ő projektben betöltött szerepükkel. Minden egyes tagunk munkája kulcsfontosságú szerepet játszott abban, hogy a Minerva platform sikeresen megszülethessen és elérhesse célját.

Az „Készítők” rovat alatt találhatja a csapat konzulensét, kiről szintén olvashat egy kis ismertetőt.

Az oldal alján megtalálhatja csoportképünket.

### 1.4.3. Regisztráció



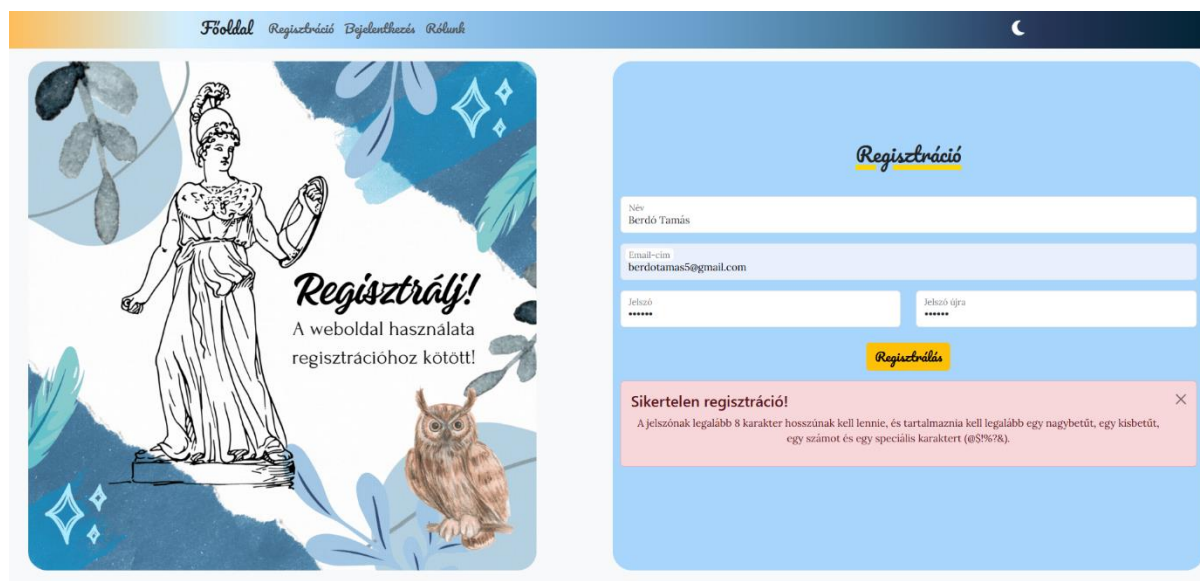
The screenshot shows a web application interface for registration. The top navigation bar includes links: [Főoldal](#), [Regisztráció](#), [Bejelentkezés](#), and [Rólunk](#). The main content area is split into two panels. The left panel has a blue and white artistic background with a classical figure and an owl, and the text: **Regisztrálj!**  
A weboldal használata regisztrációhoz kötött! The right panel is a light blue box titled **Regisztráció** containing a registration form with the following fields:  Név,  Email-cím,  Jelszó, and  Jelszó újra. A yellow **Regisztrálás** button is at the bottom.

7. ábra: Regisztrációs felület

A „Regisztráció” aloldal jobb oldalán egy regisztrációs űrlapot láthat, ahol a neve, email címe és jelszava megadása és megerősítése után regisztrálhat az alkalmazásra, ezzel teljes hozzáférést nyerhet az ingyenes eszközeinkhez. A „Regisztráció” gombra kattintva amennyiben minden adatot helyesen adott meg, egy megerősítő emailt fog kapni, amivel aktiválhatja fiókját.

Amennyiben túl gyenge jelszót adott meg, vagy az adott levelező címet már regisztrálták, esetleg egyéb hiba történt, arról egy hasonló tájékoztató hibaüzenetet fog kapni mint az a lenti ábrán látható.

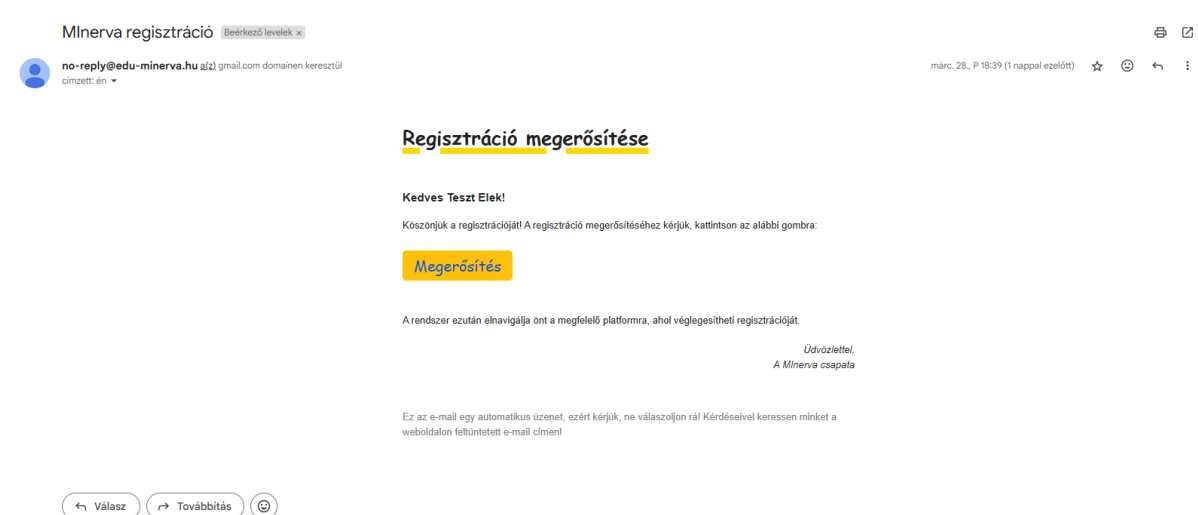




The screenshot shows a web application interface. On the left, there is a decorative illustration of a classical figure (Minerva) and an owl, with the text "Regisztrálj! A weboldal használata regisztrációhoz kötött!". On the right, there is a registration form titled "Regisztráció". The form contains fields for "Név" (Name) with the value "Berdó Tamás", "Email-cím" (Email address) with the value "berdotamas5@gmail.com", "Jelszó" (Password) with masked characters, and "Jelszó újra" (Repeat password) with masked characters. A yellow "Regisztrálás" button is below the form. Below the button, a pink error message box states: "Sikertelen regisztráció! A jelszónak legalább 8 karakter hosszúnak kell lennie, és tartalmaznia kell legalább egy nagybetűt, egy kisbetűt, egy számot és egy speciális karaktert (@\$!%&#).".

8. ábra: Regisztrációs hiba

Sikeres regisztráció után automatikusan a bejelentkező felületre navigálja az oldal, ahol a megerősítés után be is jelentkezhet.

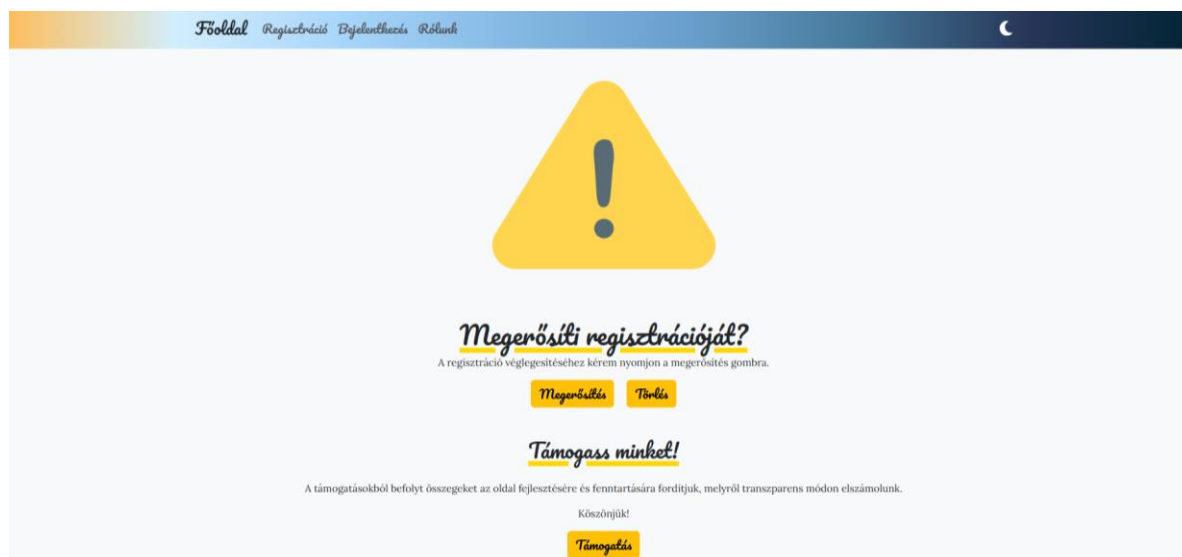


The screenshot shows an email interface. The header includes "Minerva regisztráció" and a "Beérkező levelek" button. The email is from "no-reply@edu-minerva.hu" and is titled "Regisztráció megerősítése". The body of the email says: "Kedves Teszt Elek! Köszönjük a regisztrációját! A regisztráció megerősítéséhez kérjük, kattintson az alábbi gombra." followed by a yellow "Megerősítés" button. Below the button, it says: "A rendszer ezután elnavigálja önt a megfelelő platformra, ahol véglegesítheti regisztrációját." and "Üdvözzel, A Minerva csapata". At the bottom, there is a note: "Ez az e-mail egy automatikus üzenet, ezért kérjük, ne válaszoljon rá! Kérdéseivel keressen minket a weboldalon feltüntetett e-mail címen!" and navigation buttons "Válasz" and "Továbbítás".

9. ábra: Regisztrációs email

Az levélben kapott „Megerősítés” gombra kattintva, egy regisztrációt megerősítő oldalra fogja navigálni a program, ahol eldöntheti, hogy megerősíti vagy törli a regisztrációját.

Amint ezen az oldalon rányom a „Megerősítés” gombra, a fiókja aktívvá válik, innentől bármely eszközön egyszerűen bejelentkezhet.



10. ábra: Regisztrációt megerősítő oldal

Ha az Ön e-mail címével valaki más próbált meg regisztrálni az oldalra, ezen az oldalon törölheti a regisztrációját a rendszerünkben. Természetesen később újra megpróbálhat regisztrálni a felhasznált adatokkal. Azonban egy email címhez csak egy regisztrált felhasználói fiók társítható!

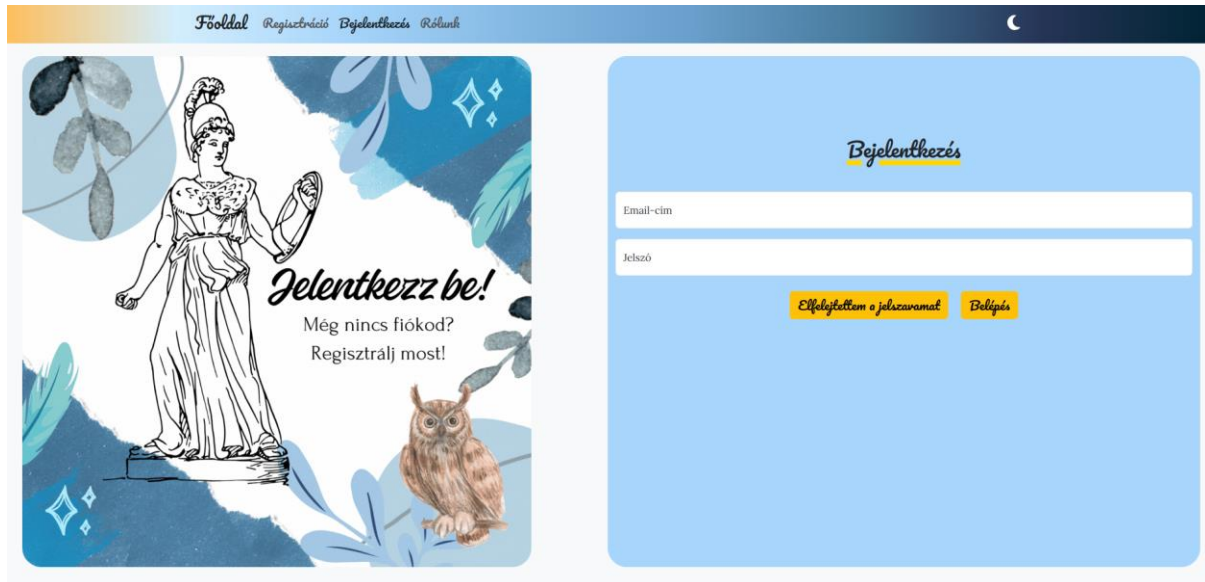
Nevét, jelszavát, email címét és minden egyéb bizalmas adatát az európai jogszabályoknak megfelelően, titkosítva saját adatbázisunkban tároljuk. A regisztrálással automatikusan elfogadja, illetve tudomásul veszi az adatvédelmi szabályainkat. Minden további információt megtalál a weboldalunk lábrészén található adatvédelmi tájékoztató oldalon.

Ha a regisztrációt egyszer már megerősítette vagy törölte, a következő hibaüzenettel fog találkozni: „Ezt a regisztrációs kódot már megerősítették vagy törölték!”.

Amennyiben nem Ön erősítette meg a kódot, vagy eltulajdonították fiókját, vegye fel a kapcsolatot az ügyfélszolgálattal az erre a célra létrehozott e-mail címen ([support@edu-minerva.hu](mailto:support@edu-minerva.hu)) vagy a hivatalos EduMinerva Discord szerveren. (<https://dc.edu-minerva.hu>).

#### 1.4.4. Bejelentkezés

A „Bejelentkezés” aloldal jobb oldalán találja a bejelentkezéshez szükséges űrlapot. Az email cím és jelszó páros megadása után kattintson a „Belépés” gombra. Amennyiben helyesen adta meg az adatait, megerősítette regisztrációját és az oldal nem jelez hibát, az oldal belépteti a felhasználót és automatikusan átnavigálja a főoldalra.



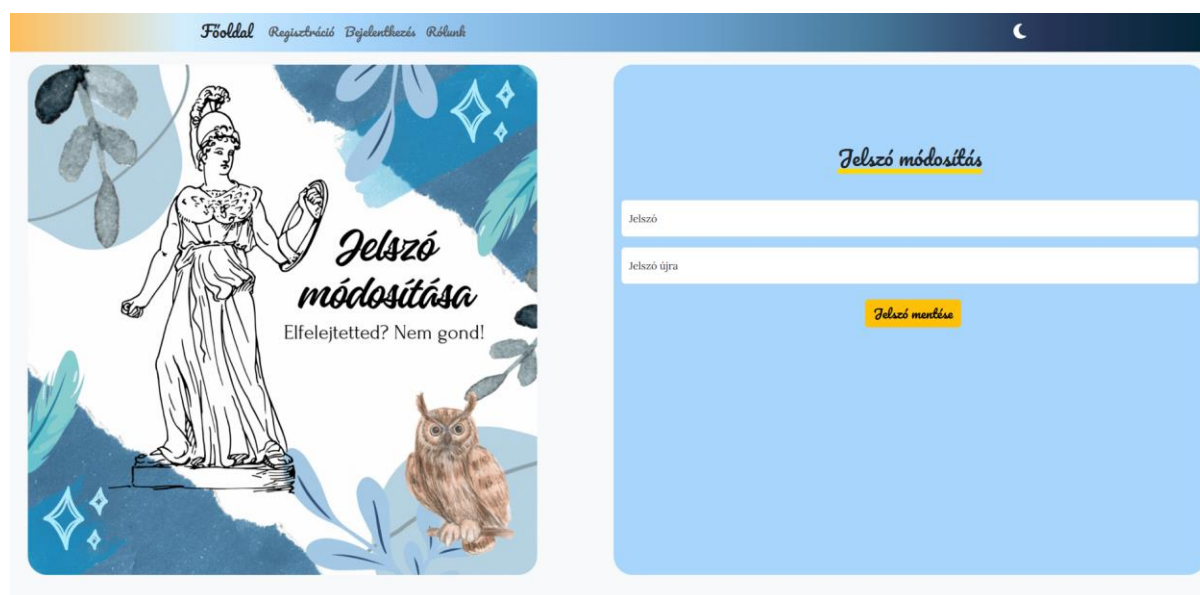
11. ábra: Bejelentkezés oldal

Elfelejtett jelszó visszaállítása: A regisztrációkor megadott email címét írja be a „Bejelentkezés” űrlap „Email-cím” mezőjébe, és kattintson az „Elfelejtettem a jelszavam” gombra, ezek után kapni fog egy helyreállító emailt, amiben talál egy „Új jelszó” gombot, amely átirányít egy aloldalra, ahol megadhatja az új jelszavát, majd az újonnan megadott jelszóval hozzáférhet a fiókjához.



12. ábra: Jelszó visszaállító email

Fontos, hogy a levélben kapott linket, amelyre a gomb mutat, ne küldje és ne mutassa meg másoknak, ugyanis ennek a hivatkozásnak a birtokában bárki állíthat be jelszót az adott fiókhoz! Azonban ez egy egyszer használatos link, amint sikeresen megadta új jelszavát ez inaktívvá válik, így nem kell félni, hogy véletlenül illetéktelen kezekbe kerül. Illetve új email kérése esetén az előzőben található hivatkozás szintén inaktívvá válik.



13. ábra: Jelszó visszaállító oldal

Ha a megadott cím nem szerepel a rendszerünkben, vagy valamilyen egyéb okból kifolyólag nem sikerült elküldenünk az emailt, azt egy hibaüzenet formájában jelzi az oldal a gombok alatt látható üres részen.

A bejelentkezés után hozzáférhet eddig rejtett elemekhez, menüpontokhoz, személyre szabhatja profilját, indíthat beszélgetéseket és írhat a fórumra.

Minden sikeres bejelentkezés után kapni fog egy emailt, ahol ellenőrizheti, hogy valóban ön lépett-e be a fiókjába.



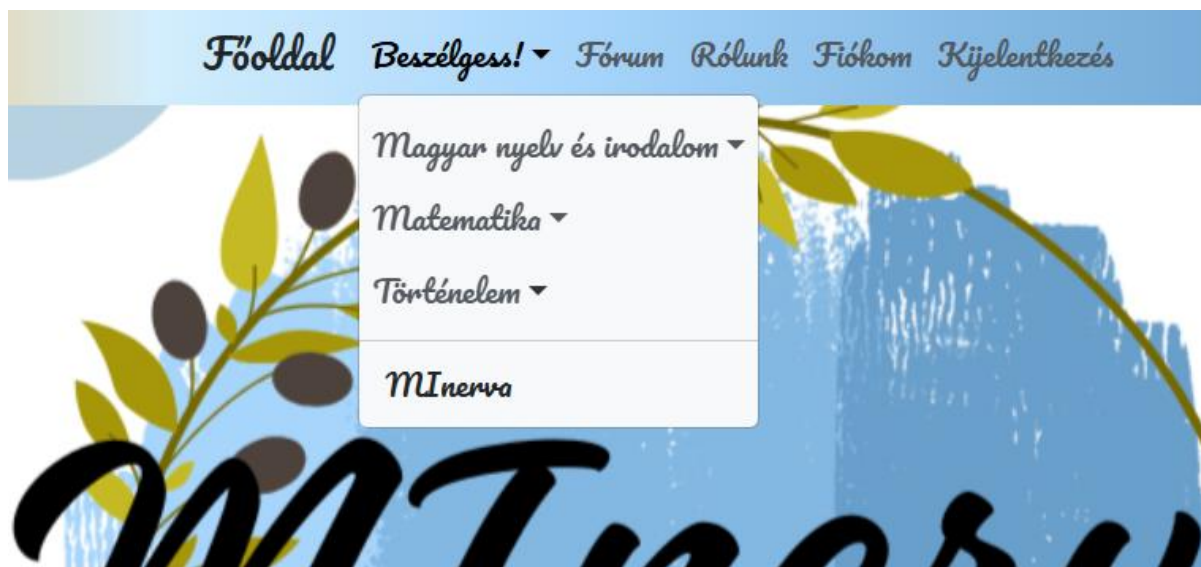
14. ábra: Bejelentkezés új eszközről

A levél tartalmazza, hogy mikor történt a belépés, az eszköz típusát, operációs rendszerét, böngészőjének nevét és IP címét.

Ha nem ön jelentkezett be a fiókjába, változtassa meg a jelszavát, és ha szükségesnek érzi vegye fel a kapcsolatot a MInerva csapattal emailben vagy a hivatalos Discord szerverünkön!

### 1.4.5. Beszélgess!

A „Beszélgess!” menüpont elérhetővé válik a bejelentkezést követően, erre kattintva megnyílik egy legördülő menü, amely tantárgyakra bontva tartalmazza az elérhető mesterséges intelligencia alapú személyeket, illetve MInervát, aki a weboldallal kapcsolatos kérdésekre igyekszik válaszolni, és általános problémákra nyújt megoldást.

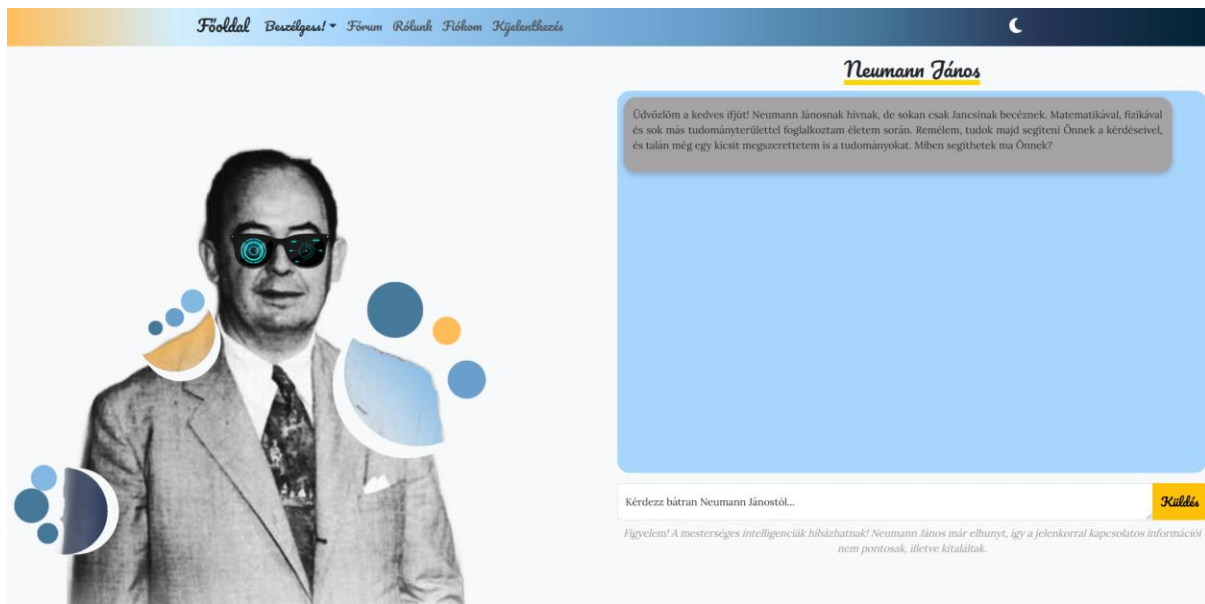


15. ábra: Beszélgess! menüpont

A listában található emberek csak az adott témakörrel és tantárggyal kapcsolatban rendelkeznek biztos tudással, illetve képesek mesélni a saját életükről és különböző múltbéli eseményekről.

Szerepjáték és egyéb csevegés béli játékokat is képesek folytatni, azonban a jövőben képesek lesznek automatikusan kezdeményezni, témaköröket, naphoz vagy valamilyen eseményhez kötött beszélgetéseket indítani.

Egy tetszőleges személy csevegőfelületét megnyitva, az oldal bal részén talál egy illusztrációt az adott személyről, jobb oldalt pedig a chat ablakot, amelyben a betöltést követően köszönti a felhasználót a mesterséges intelligencia.



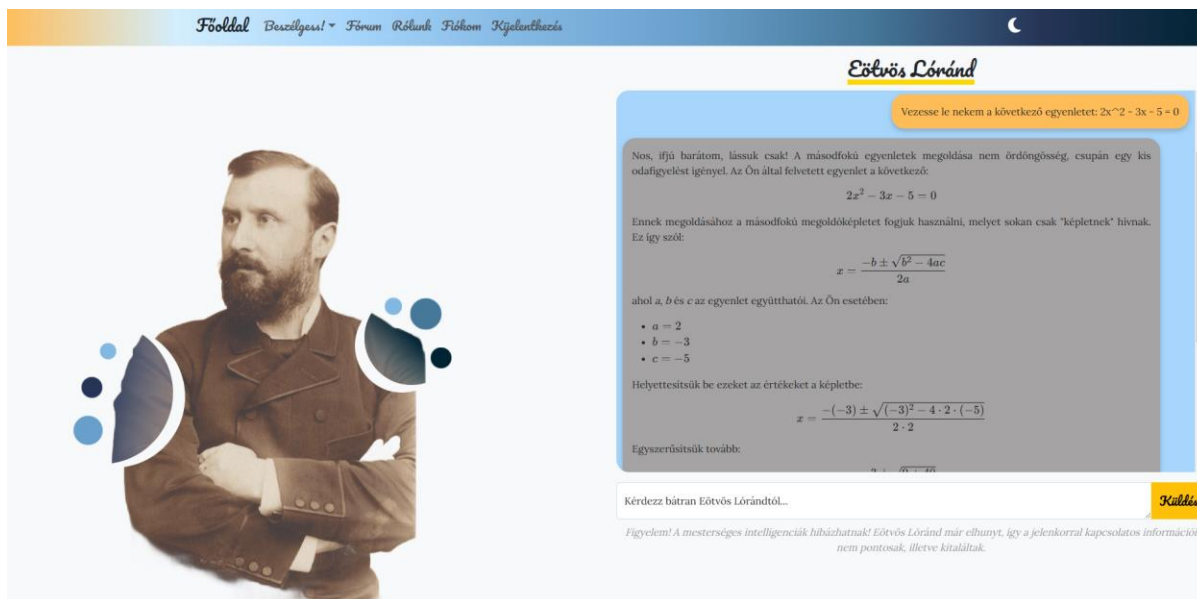
16. ábra: Beszélgetés kinézete

Az itt lefolytatott beszélgetéseket nem menti el a rendszerünk, csak addig amíg az adott oldalt megnyitva tartja a felhasználó, amint ezt bezárja vagy másik chat felületre navigál, a csevegés előzményei elvesznek, és a modell sem fog emlékezni rá.

Azonban a weboldal rendelkezik beépített szűrőkkel és védelmekkel, mint például a trágár, vagy nem odaillő szavakat nem engedi használni a rendszerünk, illetve spam gyanús üzenetek esetén a felhasználót korlátozhatja, akár ideiglenesen eltilthatja az oldal használatától.

A mesterséges intelligencia modelljeink több hasznos képességekkel is rendelkeznek, segíthetnek érthetően elmagyarázva és átláthatóan levezetni egy egyenletet, vagy adott esetben összegyűjteni, rendszerezni egy mű tartalmát.





*Fórdal Beszélgetés! Fórum Rólunk Rólunk Kijelentkezés*

**Eötvös Lóránd**

Vezesse le nekem a következő egyenletet:  $2x^2 - 3x - 5 = 0$

Nos, ifjú barátom, lassuk csak! A másodfokú egyenletek megoldása nem ördögösség, csupán egy kis odafigyelést igényel. Az Ön által felvetett egyenlet a következő:

$$2x^2 - 3x - 5 = 0$$

Ennek megoldásához a másodfokú megoldóképletet fogjuk használni, melyet sokan csak "képletnek" hívnak. Ez így szól:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

ahol  $a$ ,  $b$  és  $c$  az egyenlet együtthatói. Az Ön esetében:

- $a = 2$
- $b = -3$
- $c = -5$

Helyettesítsük be ezeket az értékeket a képletbe:

$$x = \frac{-(-3) \pm \sqrt{(-3)^2 - 4 \cdot 2 \cdot (-5)}}{2 \cdot 2}$$

Egyszerítsük tovább:

Kérdezze bátran Eötvös Lórándtól...

**Kérdés**

Figyelem! A mesterséges intelligenciák hibázhatnak! Eötvös Lóránd már elhunyt, így a jelenkorral kapcsolatos információi nem pontosak, illetve kitaláltak.

17. ábra: Egyenlet levezetés bemutatása

Az egyenleteket úgynevezett „inline” matematikai szintaxissal adhatjuk meg, amit a következőképpen írhatunk be:

- Szorzás: \*
- Osztás: / vagy ÷
- Hatványozás, négyzetre emelés: ^2.
- Hatványozás, köb: ^3.
- Tetszőleges kitevő: ^ (Alt Gr + 3 billentyű kombináció).
- Gyökvonás: √

A megadott egyenlet ezen karakterek segítségével, például így nézhet ki, másodfokú egyenlet esetén:  $2x^2 - 3x - 5 = 0$

Sajnos a magyar billentyűkiosztás rengeteg szimbólum írását nem teszi lehetővé egyszerű kombinációk segítségével, ezért más módszerekhez kell folyamodnunk. Legegyszerűbb módja ezen írásjelek bevitelére, hogy ellátogatunk egy weboldalra, ahol ezek a szimbólumok megtalálhatóak, és egyszerűen kimásoljuk onnan, majd beillesztjük a kívánt helyre.

Ilyen oldalt például a Wikipédián is találunk, ha rákeresünk, hogy „Matematikai szimbólumok listája, Wikipédia”.

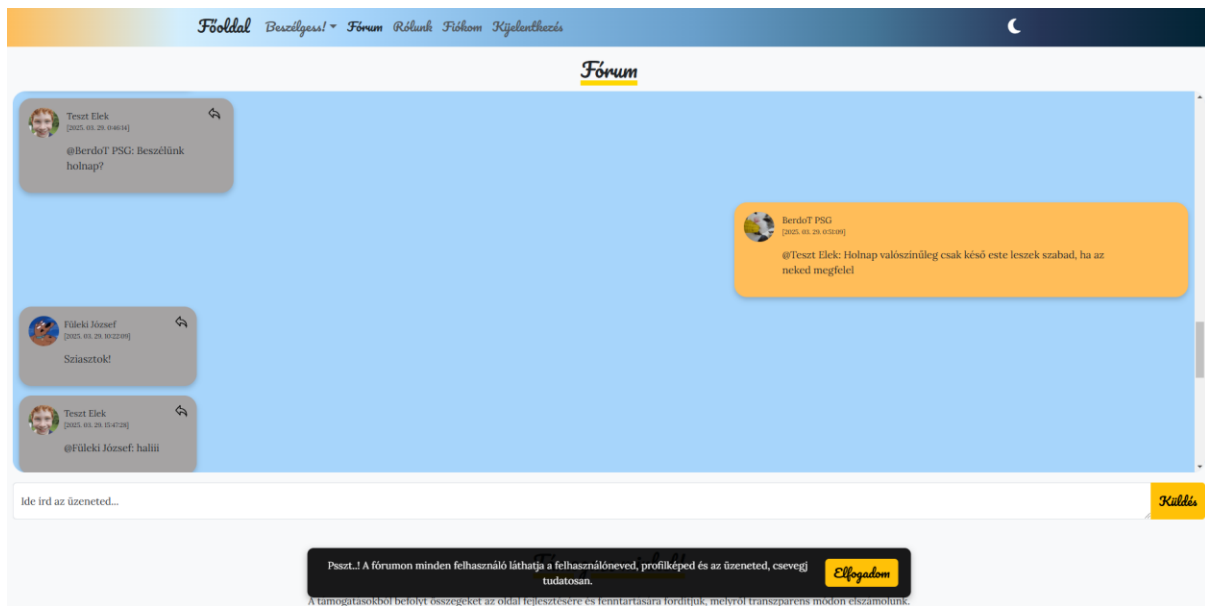
([https://hu.wikipedia.org/wiki/Matematikai\\_szimb%C3%B3lumok\\_list%C3%A1ja](https://hu.wikipedia.org/wiki/Matematikai_szimb%C3%B3lumok_list%C3%A1ja))



### 1.4.6. Fórum

A soron következő menüpont a „Fórum”, ahol a regisztrált felhasználók cseveghetnek egymással.

Az oldal betöltését követően, minden munkamenet első megnyitásakor felugrik egy figyelmeztetés, hogy a fórumra írt üzeneteket bárki láthatja, az Ön nevével és profilképével egyetemben. Ez a chat ablak szinkronizálva van a Discord szerverünkkel, ami azt jelenti, hogy az itt elküldött üzenetek megjelennek a közösségi felületünk csatornájában is.



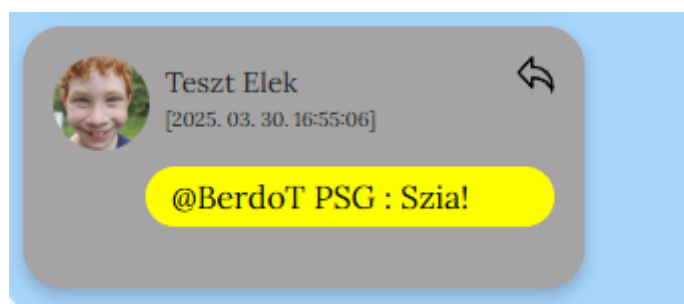
18. ábra: Fórum oldal

Rendszerünk rendelkezik egy beépített szűrővel, amely a trágár szavakat tartalmazó üzeneteket nem engedi elküldeni, illetve egy lassú üzemmóddal, ami azt jelenti, hogy egy felhasználó csak három másodpercenként küldhet egy üzenetet, elkerülve ezzel a spam üzeneteket.



19. ábra: Discord fórum szinkronizáció

A fórumon válaszolhat embereknek az üzenetük jobb felső sarkában található „Válasz” ikonnal, erre kattintva a beviteli mezőben láthatja, hogy a rendszer automatikusan megjelöli a választott felhasználót, és az így írt üzenete kiemelve fog neki megjelenni.



20. ábra: Válasz üzenet

Az elküldött üzeneteket a moderátorok törölhetik, amennyiben azt úgy ítélik meg, hogy nem helyén való tartalommal rendelkezik.

Ha egy elküldött üzenetét törölni szeretné, keressen fel egy moderátor jogosultsággal rendelkező felhasználót a Discord szerverünkön, vagy írjon egy emailt a [support@edu-minerva.hu](mailto:support@edu-minerva.hu) címre, ahol pontosan megadja melyik üzenetét szeretné eltávolítani.

A moderálás működése jelenleg a következő: A Discord szerveren gondosan kiválasztott felhasználók rendelkeznek üzenet törlési jogosultsággal, amint ezen platform „fórum” csatornájából törölnek egy üzenetet, az az integrált applikáción keresztül automatikusan törlődik a MInerva webes fórum felületéről is.

### 1.4.7. Fiókom

A navigációs sáv utolsó előtti menüpontja a „Fiókom”, itt kezelheti személyes adatait, állíthatja be nevét, iskoláját, országát, nyelvét és osztályát.



21. ábra: Fiókom oldal

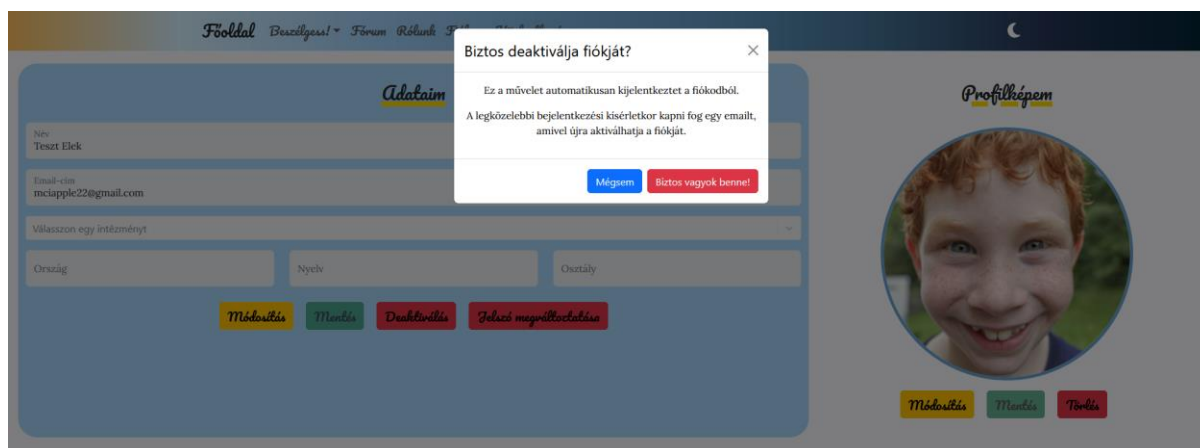
Adatai szerkesztéséhez kattintson a sárga „Módosítás” gombra az oldal bal oldalán elhelyezkedő űrlapon, ekkor módosíthatóvá válnak az információit tartalmazó beviteli mezők.

Az iskolája módosításához kezdje el gépelni az intézmény nevét, vagy írja be a települést, ahol elhelyezkedik az iskola, ekkor kilistázza az összes rendszerben lévő helyi intézményt.

A „Mentés” gomb megnyomása után az űrlap alján megjelenik egy visszaigazoló értesítés, amennyiben a mentés sikeres volt, a módosítások egyből életbe is léptek, azonban, ha a rendszer hibát észlelt, illegális karaktereket vagy trágár szavakat érzékelt a beállított adatai között, a mentés sikertelen lesz.

Profilképét módosíthatja az alatta található „Módosítás” gombbal, ekkor megnyílik a rendszere fájl választó programja, ahol egy kép kiválasztása után az meg is jelenik az oldalunkon, ezt ne felejtse el véglegesíteni a zöld „Mentés” gombbal. Amennyiben ez sikeresen megtörtént, kapni fog egy értesítést a bal oldali űrlap alján. Sikertelen mentés esetén próbálja újra később, vagy egy másik képpel. Fontos tudnivaló, hogy a feltölteni kívánt fájl nem lehet nagyobb, mint 6megabájt, azonban a legtöbb formátumot kezeli az alkalmazásunk, így például lehetséges mozgó, „.gif” kiterjesztésű képet beállítani.

Jelszavát szintén ezen az oldalon, a piros „Jelszó megváltoztatása” gombra kattintva módosíthatja, ekkor felugrik egy ablak, ahol megerősítheti döntését. Ez esetben kap egy emailt, ami elnavigálja önt egy oldalra, ahol megadhatja új jelszavát, sikeres mentést követően az új jelszó használatával fog tudni belépni fiókjába.



22. ábra: Fiók oldal, felugró ablak

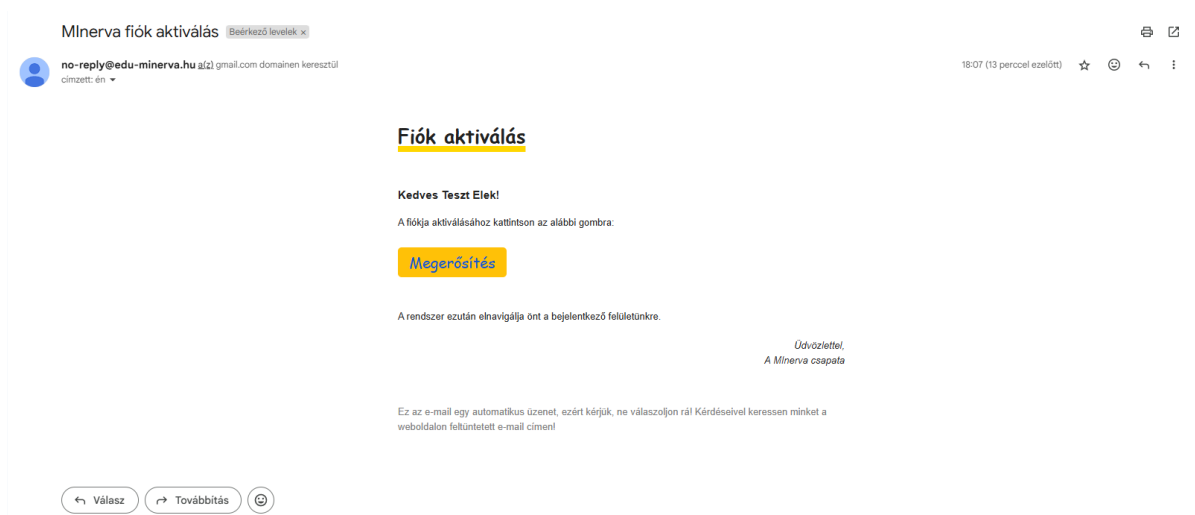
Fiókja deaktiválását az alábbi piros „Deaktiválás” gombra kattintva teheti meg, ekkor felugrik egy megerősítő ablak, ahol eldöntheti, hogy valóban szeretné-e deaktiválni a profilját.

Amennyiben megerősítette választását, az oldal automatikusan kijelentkeztette, ahogy azt a felugró figyelmeztetésben is részleteztük.

Amikor egy fiókot deaktiválnak, az technikailag elérhetetlenné válik más felhasználók számára, és a rendszerünk is annak tekinti, így például a fórumon nem jelenik meg továbbá a felhasználóneve, profilképe és nem kap promóciós vagy bármely típusú értesítéseket, leveleket, kivételt képez ez alól a Discord szerverünk „fórum” csatornája, ahol továbbra is láthatóak maradnak ezek az adatok. Amennyiben ezeket törölni szeretné, vegye fel a kapcsolatot egy moderátorral, vagy írjon a [support@edu-minerva.hu](mailto:support@edu-minerva.hu) címre.

Fiókját bármikor újra aktiválhatja, ehhez csak nyissa meg a „Bejelentkezés” oldalunkat, és töltsse ki email és jelszó kombinációját, amennyiben helyesen adta meg ezen adatokat, kapni fog egy megerősítő emailt, amelyben egy „Megerősítés” gombot talál. Erre kattintva fiókja újra aktív állapotba kerül, innentől újra használhatja és újra megjelenik rendszerünk minden részében.

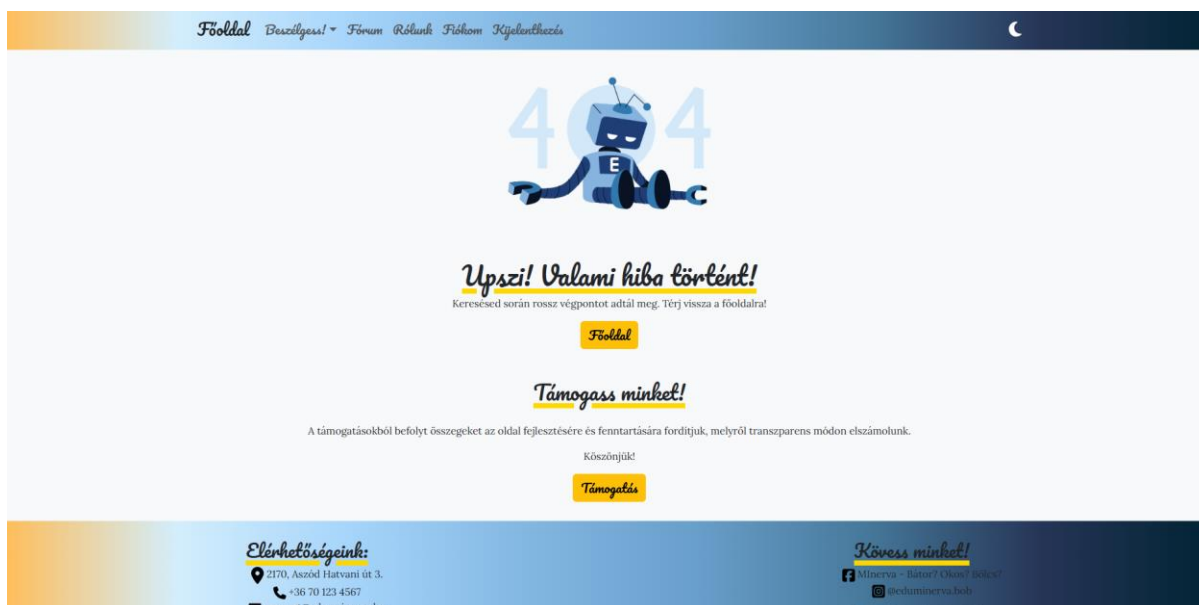
Amennyiben nem szeretné aktiválni fiókját, NE kattintson rá a „Megerősítés” gombra, és törölje az emailt. Ha később újra be szeretne jelentkezni, ismét küldünk egy aktivációs levelet.



23. ábra: Újra aktiválás megerősítő email

### 1.4.8. Az oldal nem található

Ha egy olyan hivatkozásra téved, amely már nem működik, vagy félre gépelt egy linket, ez az oldal fogja fogadni. Innen könnyedén megkeresheti a keresett oldalt a navigációs sáv segítségével.



24. ábra: 404 az oldal nem található

Ha egy védett, csak bejelentkezéssel elérhető menüpontot próbál elérni, de a jelenlegi munkamenete lejárt vagy nem volt bejelentkezve, a rendszer automatikusan a „Bejelentkezés” oldalra fogja navigálni.

## 2. Fejlesztői dokumentáció

### 2.1. Témaválasztás

Az elmúlt években felkapottá vált a mesterséges intelligencia és a nagy nyelvi modellekkel való tanulás, azonban a legtöbb ingyenesen használható megoldás általános felhasználásra, javarészt az interneten talált adatokkal került betanításra, ezért gyakran pontatlan és téves információkkal szolgálnak. A projektünk elkészítése előtt, készítettem egy felmérést, ahol az akkor legfejlettebb ingyenesen elérhető GPT-4 modellt használva többszörösen kitöltöttem hetedik, nyolcadik osztályos magyar nyelv és irodalom tesztfeladatokat, melyek eredménye 65% és 75% közé esett. Céлом, egy olyan nyelvi modell létrehozása, illetve finomhangolása, amely ennél sokkal pontosabban és megbízhatóbban működik, mind ezt egy biztonságos és barátságos környezetben. Az alapvető ötletet egy Character AI nevű alkalmazás inspirálta, ami egy ChatGPT alapú beszélgetős alkalmazás, ahol többnyire videójáték karakterekkel lehet társalogni, szerepjátékozni. Azonban ez az alkalmazás teljes mértékben a mesterséges intelligencia kreativitására támaszkodik, épp ezért nem érdemes valós, hiteles információt várni tőle, ellenben a mi projektünkkel, ahol pontos és hihető adatokkal láttuk el a nyelvi modellt.

Mindezek mellett mindig is szerettem volna látni, hogy hogyan működik egy ehhez hasonló közösségi alkalmazás felépítése, programozási háttere, épp ezért én strukturáltam és terveztem meg az egész backend felépítését. Külsős modulok használata helyett igyekeztem sajátokat létrehozni, hogy jobban megértsem azok működését, ilyen például a hitelesítő modul, a „jsonwebtoken”, amit teljes mértékben én írtam újra, az interneten található működési elvek és algoritmusok alapján.

## 2.2. Alkalmazott fejlesztői eszközök

### 2.2.1. Programok, szoftverek, alkalmazások:

- **Visual Studio Code**

Az alkalmazás teljes forráskódját ebben a szerkesztőben készítettem a könnyű kezelhetősége és bővíthetősége miatt.

- **DB Browser for SQLite**

Az adatbázishoz használt grafikus megjelenítő és szerkesztő eszköz.

- **Microsoft Office Word**

A dokumentáció szerkesztéséhez használtam.

- **Gimp 2.10**

A Dokumentációban látott képek szerkesztésére használtam.

- **Github Desktop**

A git verziókezeléshez használt egyszerű grafikus program.

- **Git Bash**

A git verziókezeléshez használt parancssor alapú program.

- **Bun.sh**

A backend a Bun all-in-one eszköztárra épült. Ez tartalmazza az adatbázis kezelőt, csomagkezelőt és a TypeScript fordítót is.

- **NodeJS**

A frontend alapjául szolgáló JavaScript motor.

- **NPM**

A NodeJS alapértelmezett csomagkezelő rendszere.

- **Postman**

A REST API kérések egyszerű tesztelésére és hibakereséséhez használtam.

- **Brave böngésző**

Egy Chromium alapú böngésző, főként ebben teszteltem az alkalmazást.

- **Firefox böngésző**

Egy nyílt forráskódú webböngésző, amely nem Chromeium alapú, ebben is teszteltem az alkalmazásunkat.

- **Cloudflare**

A Cloudflare egy hálózati szolgáltatásokat nyújtó eszköz, ami biztosítja a levelező rendszerünkben használt egyedi címeket, ezekre érkező levelek továbbítását a gmail rendszerébe, domainünk DNS beállításait, túlterhelés elleni védelmet és a biztonságos HTTP kapcsolatot.

- **Google Gmail**

A Google levelező rendszerét használtam az emailek kezelésére, és a levelező rendszerhez szükséges biztonsági kulcsokat is a Gmail szolgáltatja.

- **Google Gemini / AI studio**

A Gemini AI platformja biztosítja az oldal mögött álló nyelvi modellt, ez egy akár ingyenesen, kereskedelmi felhasználásra is használható alkalmazásprogramozási interfész.

- **OpenSSL**

Az adatok titkosításához szükséges privát és publikus kulcspár generálásához használtam.

- **dbdiagram.io**

Az adatbázis kapcsolatok vizualizálására használatos eszköz.

- **Discord:**

A Discord felületén tartottuk az online megbeszéléseket, hanghívásokat.

- **Messenger**

A gyors írásbeli kommunikációt a Messenger csoportunk segítette.

- **Canva**

A frontend alapjául szolgáló sablont ebben terveztük meg.



- **diagrams.net**

Ábrák rajzolása, tervezése a dokumentációhoz.

- **Discord Developer Portal**

A Discord szerveren lévő integráció és alkalmazás létrehozásához.

- **JwtSecret.com**

A JsonWebToken titkos kulcsának generálásához használtam.

- **React Bootstrap**

Ezen weboldal segítségével írtuk meg a React modulok nagy részét.

### 2.2.2. Backend Modulok:

- **Nodemailer és @types/nodemailer**

A Nodemailer modult a levelező rendszer fejlesztéséhez használtam, és letöltöttem hozzá a TypeScripthez szükséges típusokat tartalmazó könyvtárat.

<https://www.npmjs.com/package/nodemailer>

<https://www.npmjs.com/package/@types/nodemailer>

<https://www.nodemailer.com/>

- **Sharp**

A Sharp képek kezelésére, módosítására használható modul, a profilképek optimalizálására használtam.

<https://www.npmjs.com/package/sharp>

- **Discord.js**

A Discord platformmal való kommunikációhoz használtam.

<https://www.npmjs.com/package/discord.js>

<https://discord.js.org/>

- **@google/generative-ai**

A Gemini API-val való kommunikációra használható interfész.

<https://www.npmjs.com/package/@google/generative-ai>

### 2.2.3. Frontend modulok:

- **React**

A Facebook által fejlesztett frontend könyvtár, amely segítségével könnyedén építhetünk interaktív weboldalakat.

<https://www.npmjs.com/package/react>

- **React-Bootstrap**

Ez a modul tartalmazza a Bootstrap 5 komponenseket React kompatibilis formában.

<https://www.npmjs.com/package/react-bootstrap>

- **Bootstrap**

A hagyományos Bootstrap modul biztosítja a stílusokat és alapvető formázásokat.

<https://www.npmjs.com/package/bootstrap>

- **React-dom**

Ennek segítségével építhető be a React komponensei egy hagyományos html fájlba.

<https://www.npmjs.com/package/react-dom>

- **React-router-dom**

Egyszerűsíti és átláthatóbbá teszi a react-dom használatát.

<https://www.npmjs.com/package/react-router-dom>

- **React-markdown**

Ennek segítségével jelennek meg az AI chatablakokban szépen formázott szövegek, listák, matematikai problémák.

<https://www.npmjs.com/package/react-markdown>

- **Katex**

Ennek a könyvtárnak a segítségével jeleníti meg alkalmazásunk webes felületen a matematikai képleteket.

<https://www.npmjs.com/package/katex>

- **Rehype-katex**

Ennek a könyvtárnak a segítségével építi fel grafikusan a react-markdown a matematikai képleteket.

<https://www.npmjs.com/package/rehype-katex>

- **Remark-math**

A matematika szintaxist és formázást teszi lehetővé a react-markdown számára.

<https://www.npmjs.com/package/remark-math>

- **React-player**

Egy nagy kompatibilitású videó lejátszó modult biztosít minden eszközre.

<https://www.npmjs.com/package/react-player>

- **React-select**

Egy egyszerűen használható legördülő listát biztosít, amelyben keresni is lehet.

<https://www.npmjs.com/package/react-select>

- **React-scripts**

Ennek segítségével futtatható, építhető és kezelhető egy React alapú alkalmazás.

<https://www.npmjs.com/package/react-scripts>

- **Serve**

Egy React project elkészülte után, fel kell azt építeni a fentebb található react-scripts segítségével, ezek után, statikus fájlokat kapunk, amiket ezzel, a Serve modullal szolgálhatunk ki a klienseknek.

<https://www.npmjs.com/package/serve>

<https://create-react-app.dev/docs/deployment/>

## 2.2.4. Programozási nyelvek

### - TypeScript programnyelv

Ez a legalkalmasabb nyelv webes alkalmazások háttérének fejlesztésére.

Főként az előzetes ismereteim, nagy kompatibilitása, támogatottsága és jól kezelhető tipizált moduljai miatt választottam.

### - JavaScript programnyelv

Egyszerűen használható, hatalmas kompatibilitással rendelkező nyelv. Az internetes kommunikáció alapjai minden esetben erre épülnek.

## 2.2.5. Leírónyelvek, nyelvi kiterjesztések

### - HTML

A weboldalak alapvető szerkezetét adja, ebbe épülnek be a későbbi modulok, például a React.

### - CSS

Ez egy stílusleíró nyelv, oldalunk kinézetét ez határozza meg.

### - JSON

Programnyelvfüggetlen adatok strukturálására, adatátvitelre feltalált nyelv.

Ebben tároljuk az alkalmazásunk konfigurációs adatait, kulcsait, valamint az AI modellek finomhangolásához használt adatokat.

### - JSX

A JavaScript nyelv XML-hez hasonló kiterjesztése, ez lehetővé teszi, hogy a React alkalmazásunk megfelelően, átláthatóan formázzuk fejlesztés közben.

### - SQL

Relációs adatbázisokhoz használt lekérdezőnyelv, ezzel kommunikál a backend az SQLite adatbázissal.

## 2.3. Adatmodell, Adatbázis felépítés

### 2.3.1. „credentials” tábla, és felépítése:

Ebben a táblában tároljuk a felhasználók alapvető információs és hitelesítési adatait, például az email címét, felhasználónevét és jelszavának titkosított hashjét, amit egy egyirányú algoritmussal kódoltunk.

Minden felhasználó rekordja egy egyedi elsődleges 'id' azonosítót kap, mellyel a továbbiakban egyszerűen azonosíthatjuk a felhasználót az adatszerkezetben.

Az emailHash létrehozásához determinista hashelést használunk, ez azt jelenti, hogy például az „example@mail.com” kivonatértéke minden alkalommal ugyan az, így az email cím egyedisége könnyen ellenőrizhető.

Erre azért van szükség, mert minden személyes adatot, pl email cím és felhasználónév titkosítva tárolunk, így nem vizsgálhatjuk meg egyszerű szöveggént, hogy a beérkező címet tartalmazza-e már az adatbázis, hanem a két hasht hasonlítjuk össze, a nélkül, hogy minden RSA titkosított címet egyesével visszafejtenénk és megvizsgálánánk.

Mező	Adattípus és érték	Leírás
id	INTEGER PRIMARY KEY AUTOINCREMENT	A felhasználó egyedi azonosítója
emailHash	TEXT UNIQUE	Az email cím determinista kivonata
email	TEXT	Az email cím RSA titkosított kódja.
username	TEXT	A felhasználónév RSA titkosított kódja.
passHash	TEXT	A jelszó ARGON2id hashelt titkosított kódja.
timeCreated	INTEGER DEFAULT (strftime('%s', 'now'))	A profil létrehozásának UTC ideje.
isActive	NUMERIC DEFAULT FALSE	A profil állapota. Alap értéke hamis, email megerősítés után válik aktívvá.
failedAttempts	INTEGER DEFAULT 0	A sikertelen bejelentkezések száma.

lastLogin	INTEGER	A profil utolsó aktív belépésének UTC időbélyege.
role	TEXT DEFAULT 'user'	A felhasználó szerepköre, alapértelmezett értékben 'user'.
mgmtToken	TEXT	Ez a kód szükséges a profil fontosabb műveleteihez (pl.: aktiválás, jelszó helyreállítás)
twofaSecret	TEXT	A kétlépcsős hitelesítéshez szükséges felhasználói token.

### 2.3.2. „profileDetails” tábla felépítése:

Ebben a táblában a felhasználói profil részletesebb adatait tároljuk, mint például az iskolája nevét, vagy osztályát és profilképét. Természetesen ezeket a privát adatokat a jogszabályoknak megfelelően, RSA titkosítva mentjük.

Mező	Adattípus és érték	Leírás
id	INTEGER PRIMARY KEY AUTOINCREMENT	A rekord egyedi azonosítója.
country	TEXT	A felhasználó országa.
pictureBase64Url	TEXT	A profilképének base64 kódolású formátuma.
pfpBase64Urlx128	TEXT	A profilképének base64 kódolású formátuma 128px felbontásban.
lang	TEXT	Nyelvi beállítás.
institution	TEXT	Intézmény, iskola neve.
class	TEXT	Osztály neve.
credentialsId	INTEGER UNIQUE, FOREIGN KEY	A „credentials” táblához kapcsolódó azonosító kulcs.

### 2.3.3. „forumMessages” tábla felépítése:

Ebben a táblában tároljuk a fórumra elküldött üzeneteket, és azokhoz kapcsolódó információkat, például az üzenet időbélyegét.

Mező	Adattípus és érték	Leírás
id	INTEGER PRIMARY KEY AUTOINCREMENT	Az üzenet egyedi azonosítója.
message	TEXT	A felhasználó fórumra küldött üzenete.
timeSent	INTEGER	Az üzenet időbélyege.
messageIdDiscord	TEXT	A Discord platformon megjelenő üzenet azonosítója.
credentialsId	INTEGER UNIQUE, FOREIGN KEY	A „credentials” táblához kapcsolódó azonosító kulcs.

A fórumra küldött üzenetek nincsenek titkosítva, ugyanis ezek a rendszerünkön belül publikusan küldött üzenetek, illetve egy esetleges adatbázis szivárgás esetén a felhasználó nem azonosítható egyértelmű adatok alapján, mint például név, email cím vagy intézmény név, ugyanis ezek továbbra is titkosítva vannak.

### 2.3.4. „sqlite\_sequence” tábla:

Ezt a táblát automatikusan hozta létre az adatbázis kezelő szoftverünk, ugyanis ebben tároljuk, hogy az adott táblák „autoincrement” kulcsai éppen hol járnak.

Ez egy speciális tábla, amely mezőinek nincs megadott értéke, azonban láthatólag az adatbázis TEXT és NUMERIC adatokat tárol benne.

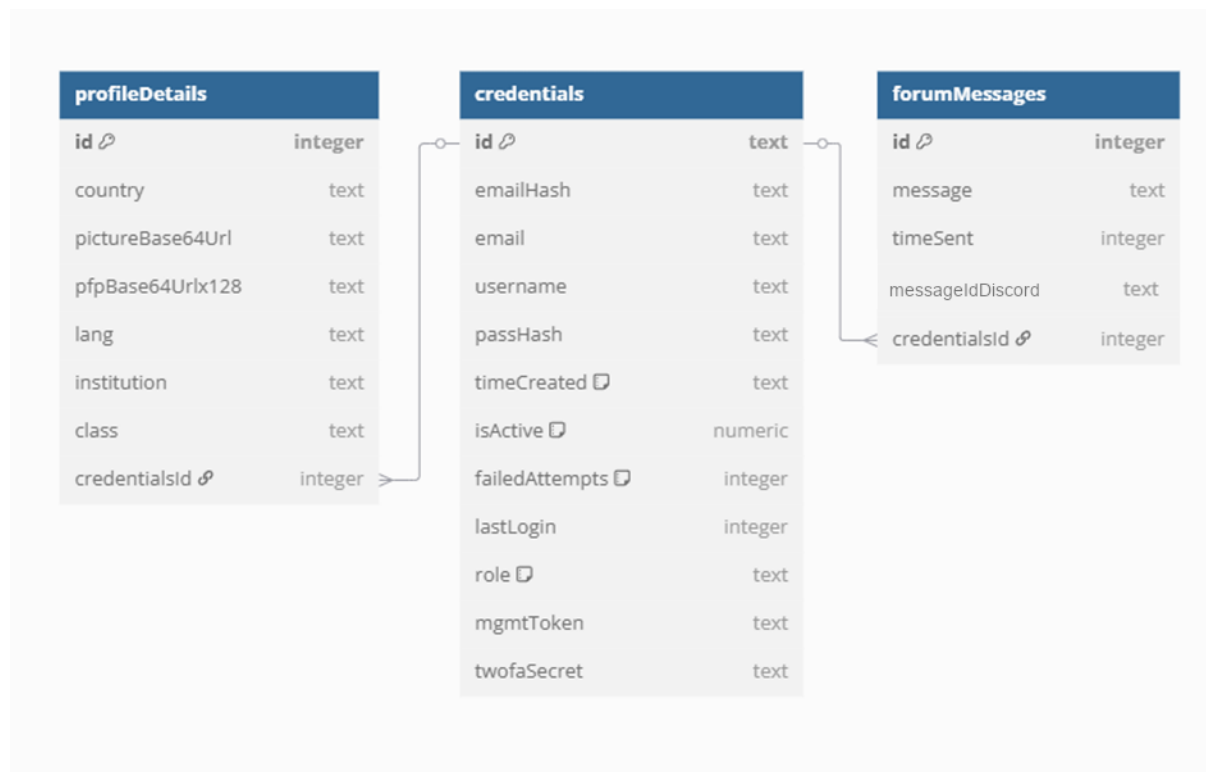
Mező	Adattípus és érték	Leírás
name	*nincs*	A számolt tábla neve.
seq	*nincs*	Az adott tábla szekvenciája.



### 2.3.5. Adatbázis táblák között lévő kapcsolat:

A „profileDetails” tábla ON DELETE CASCADE hivatkozással kapcsolódik a „credentials” táblához, azaz, ha az utóbbi táblából törölünk egy rekordot, mind két táblában törlődnek a kapcsolódó adatok.

A „forumMessages” tábla ugyan úgy a „credentials” tábla „id” mezőjéhez kapcsolódik, azonban ezen táblából nem törlődnek a felhasználó üzenetei, ha az törli magát a rendszerünkben. Ez esetben az üzenethez nem fog tartozni felhasználónév, így annak szerzője [Inactive user]-ként fog megjelenni a fórumon.



25. ábra: Táblák közötti kapcsolat - <https://dbdiagram.io/>

## 2.4. Részletes feladatspecifikáció, algoritmusok, forráskódok

A következőkben részletesen bemutatok néhány kódrészletet és algoritmust, azok működését és alapelvét, azonban a teljes forráskód publikusan elérhető a GitHub oldalamon.

<https://github.com/mcitomi/minerva>

### 2.4.1. Titkosító funkció:

```
27 export function encryptRSA(text: string) { // A publikus kulccsal lekódoljuk a szöveget,
    maximum 374 karakter (RSA-3072) és base64 kódolásban visszaadjuk
28     try {
29         const publicKey = readFileSync(join(process.cwd(), "secrets", "public_key.pem"),
            "utf-8");
30
31         const encryptedData = publicEncrypt(
32             {
33                 key: publicKey,
34                 padding: constants.RSA_PKCS1_OAEP_PADDING, // Mivel a frontend is ilyen
                    paddingot használ, backenden is ezzel kódolunk, így használhatjuk ugyan azt a
                    functionot decryptre
                    oaepHash: "sha512"
35             },
36             new Uint8Array(Buffer.from(text))
37         );
38     } catch (error) {
39         console.error(error);
40         return undefined;
41     }
42     return encryptedData.toString("base64");
43 }
44 }
45 }
```

26. ábra: RSA titkosító algoritmus

Az ábrán látható „encryptRSA” függvény valósítja meg a felhasználók adatainak titkosítását.

Ehhez egy úgynevezett aszimmetrikus RSA titkosítást használunk, aminek működési elve a következő: Beolvassuk a „secrets” mappából a „public\_key.pem” fájlt, ezzel a kulccsal tudjuk levédeni az adatainkat, és a hozzá tartozó „private\_key.pem” fájllal pedig olvashatjuk, azaz dekódolhatjuk a titkosított üzenetet.

A titkosított adatot a beépített „node:crypto” modul „publicEncrypt” függvényével hozzuk létre, ahol átadjuk neki a publikus kulcsot, egy úgynevezett „padding”-ot, ami a kitöltési algoritmust határozza meg, ez jelen esetben „RSA\_PKCS1\_OAEP\_PADDING”, ugyanis a böngészők és a biztonsági standardok ezt ajánlják. Végül beállítjuk a hashelési algoritmust, ami jelen esetben SHA-512, ami egy biztonságosabb megoldás, mint az alapértelmezett SHA-1. A függvény második értékeként átadjuk a titkosítandó szöveget egy 8 bites egész számokat

tartalmazó tömbben, ami a szöveg ASCII / UTF-8 karakterkódjait tartalmazza, amit először egy javascript Buffer objektumból szedtünk ki, mert a buffer alapvető esetben bináris adatokat tárol.

Majd végezetül a függvényünk visszatér a titkosított adattal, amelyet base64 kódolással ad vissza a nagyobb kompatibilitás és adatbiztonság érdekében, így könnyebben tárolhatjuk az adatot adatbázisban, vagy küldhetjük REST API-n keresztül gond nélkül.

Az alkalmazásunk tartalmaz egy „decryptRSA” függvényt, ami ugyan ezt a megoldást alkalmazza, viszont a „node:crypto” „privateDecrypt” függvényét használja, aminek ha átadjuk a privát kulcsot, visszafejthetjük a titkosított adatot, amit UTF-8 kódolású szöveggént visszaad a függvény.

### 2.4.2. Email küldő funkció:

```
backend > src > modules > mail > senders > sendmail
You, 6 days ago | 1 author (You)
1 import { mail_application_password } from "../.././config.json";
2 import { createTransport } from "nodemailer";
3
4 export function sendMail(targets: string[], subject: string, text: string, html: string) {
5   try {
6     var transporter = createTransport({
7       service: 'gmail',
8       auth: {
9         user: 'eduminervahu@gmail.com',
10        pass: mail_application_password
11      }
12    });
13
14    transporter.sendMail({from: 'no-reply@edu-minerva.hu', to: targets.toString(),
15      subject: subject, text: text, html: html}, function(error, info){
16      if (error) {
17        return `Error: ${error}`;
18      } else {
19        return `Email sent: ${info.response}`;
20      }
21    });
22  } catch (error) {
23    return `Error: Application crash! ${error}`;
24  }
}
```

You, 7 days ago • workin mail system

27. ábra: Üzenet küldő funkció

Az ábrán látható „sendMail” függvény a Nodemailer könyvtár segítségével emaileket küld a megadott címzetteknek.

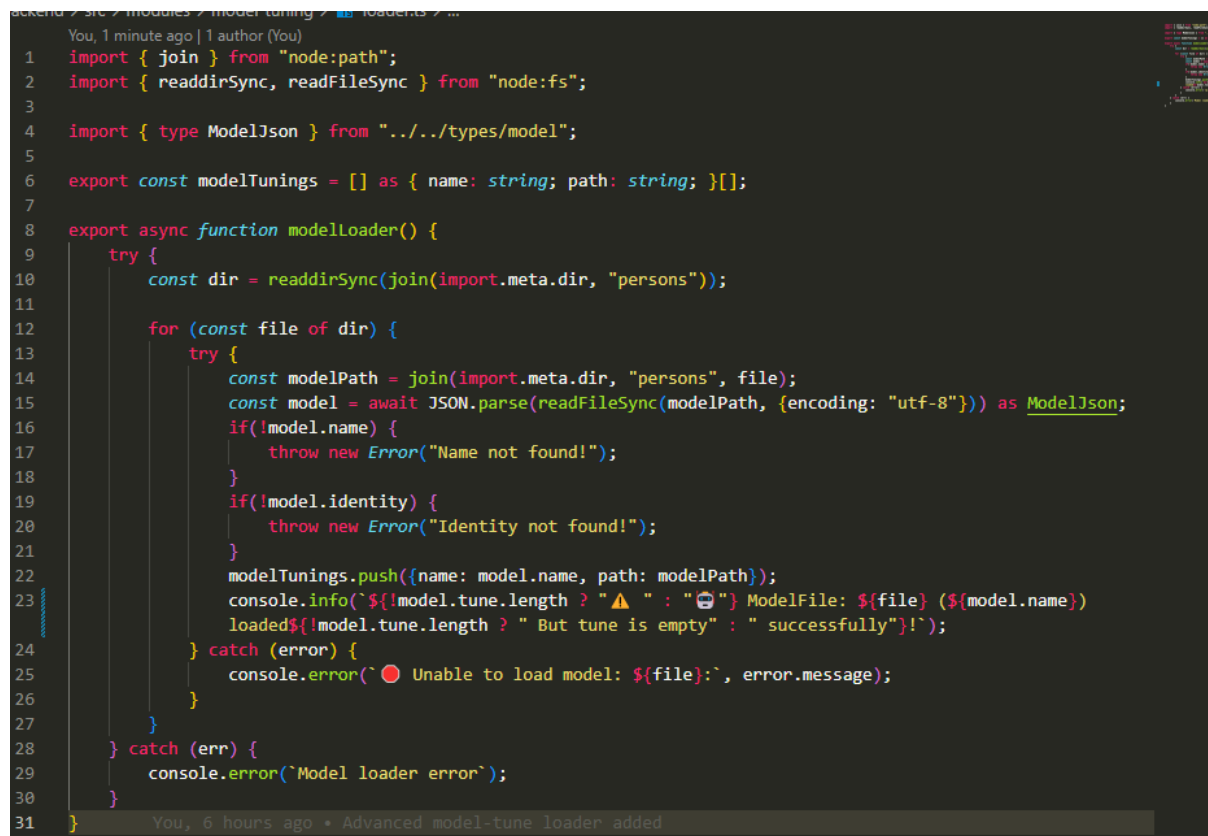
A függvénynek több paramétert adunk át, először egy szöveg tömböt, amely tartalmazza a címzettek email címeit, majd az üzenet tárgyát, üzenetét, és végül egy html kódot, ami az emailben megjelenik.

A Nodemailer „createTransport” metódusával létrehoz egy email küldőt, amely a Gmail SMTP, azaz üzenetküldő szerverét fogja használni. A hitelesítéshez a konfigurációs fájlból importált „mail\_application\_password”öt használja az eduminerva@gmail.com fiókhoz.

A „transporter.sendMail” metódus segítségével elküldjük a levelet a címzetteknek, kiknek címét a „toString()” javascript metódus segítségével alakítjuk tömbből szöveges felsorolássá.

Sikeres küldés esetén a függvény visszatér az „Email sent: ....”, szöveggel, ellenkező esetben egy „Error: ...”, szöveggel tér vissza.

### 2.4.3. Model tuning betöltő modul:



```

1  import { join } from "node:path";
2  import { readdirSync, readFileSync } from "node:fs";
3
4  import { type ModelJson } from "../../types/model";
5
6  export const modelTunings = [] as { name: string; path: string; }[];
7
8  export async function modelLoader() {
9    try {
10     const dir = readdirSync(join(import.meta.dir, "persons"));
11
12     for (const file of dir) {
13       try {
14         const modelPath = join(import.meta.dir, "persons", file);
15         const model = await JSON.parse(readFileSync(modelPath, {encoding: "utf-8"})) as ModelJson;
16         if(!model.name) {
17           throw new Error("Name not found!");
18         }
19         if(!model.identity) {
20           throw new Error("Identity not found!");
21         }
22         modelTunings.push({name: model.name, path: modelPath});
23         console.info(`${!model.tune.length ? "⚠️ " : "✅"} ModelFile: ${file} (${model.name})
24         loaded${!model.tune.length ? " But tune is empty" : " successfully"}!`);
25       } catch (error) {
26         console.error(`🔴 Unable to load model: ${file}:`, error.message);
27       }
28     }
29   } catch (err) {
30     console.error(`Model loader error`);
31   }

```

28. ábra: Tuning betöltő modul

A fentebb látható „modelLoader” függvény felelős a mesterséges intelligencia modellünk tuning beállításainak betöltéséért.

A funkciót egyből, a backend induláskor meghívjuk, ami ekkor megkeresi a „src/modules/model-tuning/persons” mappában az összes fájlt. Ebben a mappában tároljuk egy-egy személy adatait JSON formátumban, a modul ezt meg is próbálja beolvasni, amennyiben sikerül, és a fájl hibátlanul fel van töltve az adatokkal, annak a személynek a neve, és a személyhez tartozó tuning fájl elérési útvonala belekerül a „modelTunings” tömbe, amit majd a későbbiekben használhatunk API hívások során, ugyanis ez a változó fájlon kívülről is elérhető, exportált objektum.

Amennyiben sikeresen betöltött egy fájlt, vagy netán hibát érzékelt, minden információról tájékoztat minket a program konzoljában.

```
🌐 REST backend server started on port 3030
🔴 Unable to load model: ady.json: Name not found!
🔴 Unable to load model: arany.json: Name not found!
🔴 Unable to load model: babits.json: Name not found!
🔴 Unable to load model: bolyaiFarkas.json: Name not found!
⚠️ ModelFile: bolyaiJanos.json (bolyai_janos) loaded But tune is empty!
🔴 Unable to load model: erdos.json: Name not found!
🔴 Unable to load model: horthy.json: Name not found!
🔴 Unable to load model: hunyadi.json: Name not found!
🔴 Unable to load model: jozsef.json: Name not found!
⚠️ ModelFile: kolcsey.json (kolcsey_ferenc) loaded But tune is empty!
🔴 Unable to load model: kossuth.json: Name not found!
🔴 Unable to load model: koztolanyi.json: Name not found!
🔴 Unable to load model: lovasz.json: Name not found!
🔴 Unable to load model: madach.json: Name not found!
🔴 Unable to load model: matyas.json: Name not found!
🤖 ModelFile: minerva.json (minerva) loaded successfully!
⚠️ ModelFile: neumann.json (neumann_janos) loaded But tune is empty!
🤖 ModelFile: petofi.json (petofi_sandor) loaded successfully!
🔴 Unable to load model: polya.json: Name not found!
🔴 Unable to load model: rakoczi.json: Name not found!
🤖 ModelFile: saint.json (szent_istvan) loaded successfully!
⚠️ ModelFile: szechenyi.json (szechenyi_istvan) loaded But tune is empty!
🔴 Unable to load model: szemeredi.json: Name not found!
🔴 Unable to load model: turan.json: Name not found!
found!
📁 get /auth/pk loaded!
📁 get /user/profile loaded!
📁 post /gemini-models/chat loaded!
📁 post /auth/password-request loaded!
📁 post /auth/login loaded!
📁 post /auth/register loaded!
📁 post /auth/password-reset loaded!
📁 get /auth/verify-mail/check loaded!
📁 get /auth/verify-mail/link loaded!
📁 get /auth/verify-mail/remove loaded!
```

29. ábra: ModelLoader konzoli kimenet - példa

A parancssorban különböző ikonokkal jelzi a fájlok állapotát.

A pirossal jelzett fájlok még nem állnak készen a használatra, hiányosak, amíg a sárgával jelzett sorok, már használhatóak, viszont a tuning adatok hiányosak vagy hibásak.

A robot ikonnal jelzett adatok mutatják, a hibátlanul betöltött értékeket.

#### 2.4.4. Végpontokat betöltő router

A backend szerver alapja egy egyedileg írt, strukturált, végpont betöltő és kezelő rendszer, mely a „router.ts” fájlban helyezkedik el, és egy „RequestHandler” nevű osztályt tartalmaz, amelynek három főbb metódusa van, két privát tulajdonsága és egy paramétere.

Az osztály deklarálásánál paraméterként át kell adni az adatbázis modul kontrollerét, így a végpontok el tudják majd érni az adatbázist, így biztonságosan és gyorsan kezelni azt.

A következőkben futási sorrendben bemutatom az ebben található metódusok működését, alapjait.

##### I. „register” funkció.

Ezt az eljárást a program indítása után, egyszer kell csak meghívni és futtatni, ekkor az osztály belső tulajdonságának egyikében, az „endpoints” tömbben fogja tárolni a talált elérési utakat. Ennek a tömbnek a típusa egy „Endpoint” tömb, amely egy egyénileg létrehozott típus, melynek három tulajdonsága van, a „route”, „name” és „type”, ezek mind stringként kerülnek tárolásra.

Azonban fontos tudni, hogy ez a funkció hogyan, és honnan is tölti be ezeket a végpontokat, és hogyan építi fel azokat. A backend mappán belül, a `/src/routes` mappában további almappák találhatóak.

Közvetlenül a „routes” mappában találhatóak a http metódusok nevével ellátott mappák (get, post, patch, delete), ezek adják meg, hogy a bennük található végpontokat milyen metódussal lehet elérni, ezzel egy könnyen kezelhető fájl struktúrát biztosítunk a rendszerünknek.

Ezen mappákon belül pedig további mappákat vagy fájlokat találunk, ezek adják majd meg a végpont elérési útvonalát, néhány példa erre: A „`/backend/src/routes/get/teszt/helloworld.ts`” végpontot a következőképpen érhetnénk el az interneten, a „`/routes/get/`” meghatározza, hogy a kérést típusa GET, az elérési útvonal pedig az ez után fennmaradt „`/teszt/helloworld/`” lesz. A böngészőben például így érhetjük el: <http://localhost/teszt/helloworld/>.

A fentebb említett „endpoints” tömbbe ezek a végpontok lesznek betáplálva, azonban fontos észben tartani, hogy a rendszer csak a „.ts” fájlokat keresi meg és tölti be. Egy végpont betáplált adatai a következőképpen épülnek fel:

- „route”: Fizikailag a számítógépen meghatározott elérési útvonala a fájlnak.
- „name”: Ez a tulajdonság tartalmazza a kívánt elérési utat, például a „teszt/helloworld”.
- „method”: Ez pedig a metódust határozza meg, így például lehet kettő azonos nevű elérési út, ha a http metód típusa különbözik.

Azonban ez a rendszer továbbfejleszthető és könnyedén át csoportosítható, ha például minden végpont fájlnak készítünk egy tulajdonságok részleget, ahol meghatározzuk, hogy milyen elérési útvonalra hívható meg és milyen metódussal, így nem kötelező tartani a fent említett szigorú fájl struktúrát.

## **II. „listener” funkció.**

Ezt a funkció keresi és hívja meg a megfelelő elérési pontot az „endpoints” tömbből, amelyet a fentebb leírt „register” modul töltött be, majd szolgálja ki az adott kérést a kliens felé. A programunk http szerverében ezt az egyetlen metódust kell elhelyezni, visszatérni az értékével, meghívni minden beérkező kérés esetén, és átadni neki az aktuális „Request” típusú paramétert.

Ha a beérkező kérés típusa „OPTIONS”, akkor egy sablon „OK” státusszal válaszol a szerver. Minden más típusú kérést megvizsgál, hogy található-e az „endpoints” tömbben, amennyiben megtalálta, meghívja az adott végpontot, és visszatér az értékével. Ha a kért elérési út nem található 404-es hibával tér vissza a kérés. Minden egyéb nem várt hiba esetén 500-as szerver hibakóddal térünk vissza.

Minden kimenő válaszra ráhúzzuk a következő pontban bővebben kifejtett „CORS” függvényt, amellyel a kliensek biztosan fogadni tudják majd az adott kérést.

## **III. CORS funkció.**

Mivel a Bun nem rendelkezik beépített CORS middlewarrel, ezért saját magunknak kell megoldanunk ezt a problémát.

Működése egyébként nagyon egyszerű, minden beérkező kérésre a webszerver egy válasszal reagál, melynek típusa „Response”. Ennek a függvénynek ezt a válasz paramétert kell átadni, majd ez átírja és kibővíti a fejlécét a szabványos „Cross-Origin” szabályokkal, amely engedélyez minden irányú csatlakozást, majd a végén visszatér ezzel a módosított „Response” értékkel, amivel már a szerver válaszolhat a kérésekre.



### 2.4.5. Hibátlan indulás

A fentebb látható módon más hibákat, információkat és figyelmeztetéseket is a konzol felületére ír a program, így induláskor ellenőrizhető, hogy minden modul sikeresen elindult-e.

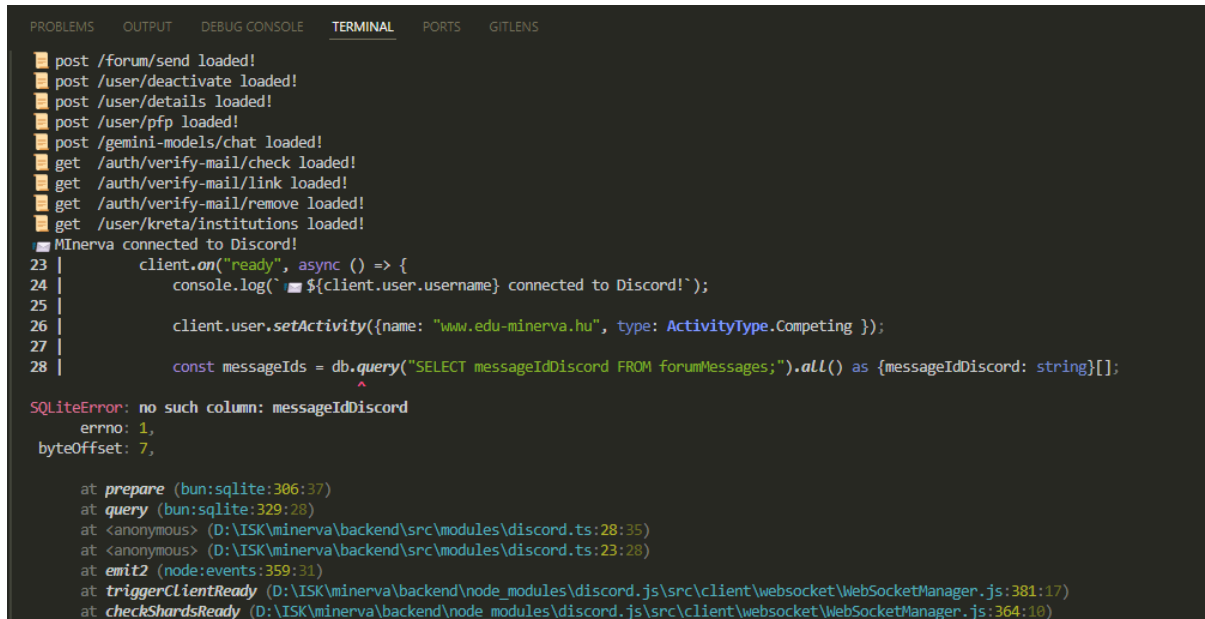
```
PS D:\ISK\minerva\backend> bun .
REST backend server started on port 3030
ModelFile: ady.json (ady_endre) loaded successfully!
ModelFile: arany.json (arany_janos) loaded successfully!
ModelFile: babits.json (babits_mihaly) loaded successfully!
ModelFile: bolyaiFarkas.json (bolyai_farkas) loaded successfully!
ModelFile: bolyaiJanos.json (bolyai_janos) loaded successfully!
ModelFile: eotvos.json (eotvos_lorand) loaded successfully!
ModelFile: erdos.json (erdos_pal) loaded successfully!
ModelFile: horthy.json (horthy_miklos) loaded successfully!
ModelFile: hunyadi.json (hunyadi_janos) loaded successfully!
ModelFile: jozsef.json (jozsef_attila) loaded successfully!
ModelFile: kolcsey.json (kolcsey_ferenc) loaded successfully!
ModelFile: konig.json (konig_gyula) loaded successfully!
ModelFile: kossuth.json (kossuth_lajos) loaded successfully!
ModelFile: kosztolanyi.json (kosztolanyi_dezso) loaded successfully!
ModelFile: madach.json (madach_imre) loaded successfully!
ModelFile: matyas.json (matyas_kiraly) loaded successfully!
ModelFile: minerva.json (minerva) loaded successfully!
ModelFile: neumann.json (neumann_janos) loaded successfully!
ModelFile: petofi.json (petofi_sandor) loaded successfully!
ModelFile: polya.json (polya_gyorgy) loaded successfully!
ModelFile: rakoczi.json (rakoczi_ferenc) loaded successfully!
ModelFile: saint.json (szent_istvan) loaded successfully!
ModelFile: szechenyi.json (szechenyi_istvan) loaded successfully!
ModelFile: turan.json (turan_pal) loaded successfully!
ModelFile: zrinyi.json (zrinyi_miklos) loaded successfully!
get /auth/pk loaded!
post /auth/login loaded!
get /forum/messages loaded!
get /forum/new loaded!
post /auth/register loaded!
post /auth/password-reset loaded!
get /user/pfp loaded!
post /auth/password-request loaded!
get /user/reactivate loaded!
get /user/profile loaded!
post /forum/profiles loaded!
post /forum/send loaded!
post /user/deactivate loaded!
post /user/details loaded!
post /user/pfp loaded!
post /gemini-models/chat loaded!
get /auth/verify-mail/check loaded!
get /auth/verify-mail/link loaded!
get /auth/verify-mail/remove loaded!
get /user/kreta/institutions loaded!
Minerva connected to Discord!
```

30. ábra: Hibátlan konzol példa

Az ábra első sorában látható, hogy a programot a „bun .” parancs segítségével elindítottam, majd a webszerver modul visszajelzett, hogy sikeresen lefoglalta és elindult a megadott port számon. Ez után láthatjuk a fentebb részletesen leírt „ModelLoader” üzeneteit, majd alatta a betöltött végpontokat, amelyt az egyéni router modul „register” függvénye írt

ki, majd végül a Discord modul jelzett vissza, hogy a „MInerva” nevű applikáció sikeresen csatlakozott a platform rendszerére.

Amennyiben egy kritikus fontosságú modul vagy program résznek nem sikerült hibátlanul elindulnia, az megállthatja az egész folyamatot. Ekkor általában a Bun keretrendszer hibakezelője veszi át az irányítást, amely leállítja a program futását, és részletes hibaösszefoglaló naplót készít, amelyet a konzolra írat.



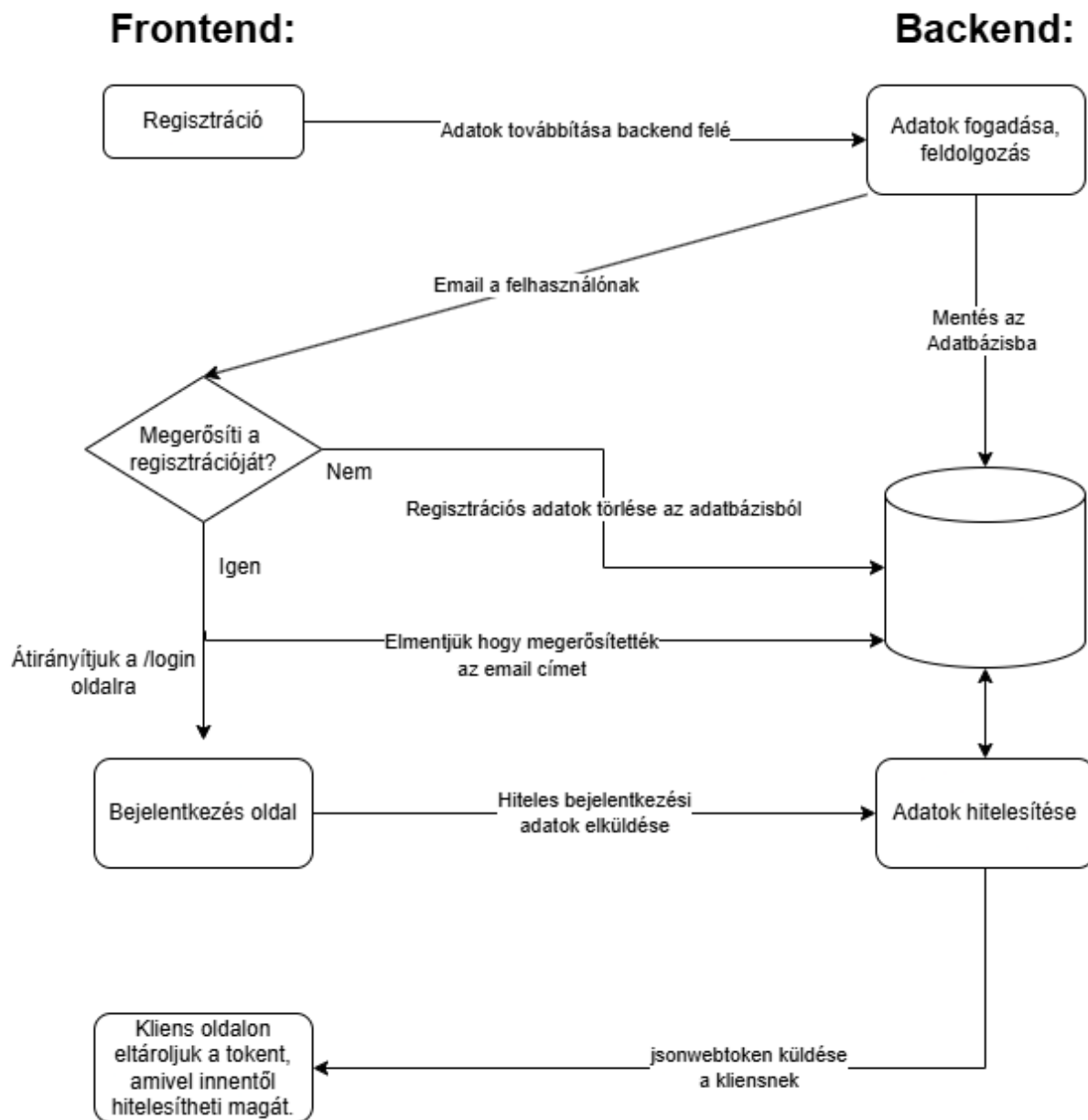
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
post /forum/send loaded!
post /user/deactivate loaded!
post /user/details loaded!
post /user/pfp loaded!
post /gemini-models/chat loaded!
get /auth/verify-mail/check loaded!
get /auth/verify-mail/link loaded!
get /auth/verify-mail/remove loaded!
get /user/kreta/institutions loaded!
MInerva connected to Discord!
23 |     client.on("ready", async () => {
24 |         console.log(` ${client.user.username} connected to Discord!`);
25 |
26 |         client.user.setActivity({name: "www.edu-minerva.hu", type: ActivityType.Competing });
27 |
28 |         const messageIds = db.query("SELECT messageIdDiscord FROM forumMessages;").all() as {messageIdDiscord: string}[];

SQLiteError: no such column: messageIdDiscord
  errno: 1,
  byteOffset: 7,

at prepare (bun:sqlite:306:37)
at query (bun:sqlite:329:28)
at <anonymous> (D:\ISK\minerva\backend\src\modules\discord.ts:28:35)
at <anonymous> (D:\ISK\minerva\backend\src\modules\discord.ts:23:28)
at emit2 (node:events:359:31)
at triggerClientReady (D:\ISK\minerva\backend\node_modules\discord.js\src\client\websocket\WebSocketManager.js:381:17)
at checkShardsReady (D:\ISK\minerva\backend\node_modules\discord.js\src\client\websocket\WebSocketManager.js:364:10)
```

31. ábra: Bun keretrendszer hibakezelés példa

Az ábrán látható hibát az okozta, hogy az adatbázisban nem volt létrehozva a „messageIdDiscord” mező, ezért a hibakezelő rendszer leállította a program futását, és kiíratta a probléma pontos forrását.

**2.4.6. Authentikációs folyamat:**

32. ábra: Authentikációs ábra

## 2.5. Tesztelési dokumentáció

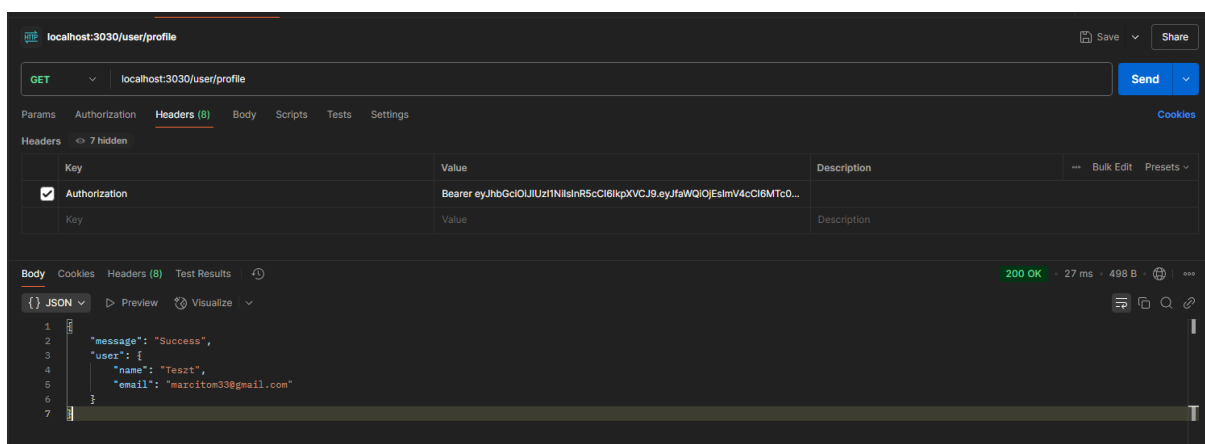
### 2.5.1. Védett hálózati végpont elérésének tesztelése:

A „/user/profile” végpont célja, hogy az autentikált felhasználó nevét, email címét és egyéb privát információit kérhetjük le. A tesztelés során ellenőrizzük, hogy a végpont megfelelően működik-e különböző felhasználói műveletek esetén, és hogy a megfelelő üzeneteket kapjuk-e vissza hibás vagy extrém bemenet esetén.

A végpont normál esetben egy GET kérést vár, egy Authorization fejléccel, és egy Bearer tokennel, ezen adatok alapján hitelesíti a felhasználót, és visszaadja a kért forrásokat.

### 2.5.2. Normál teszteset:

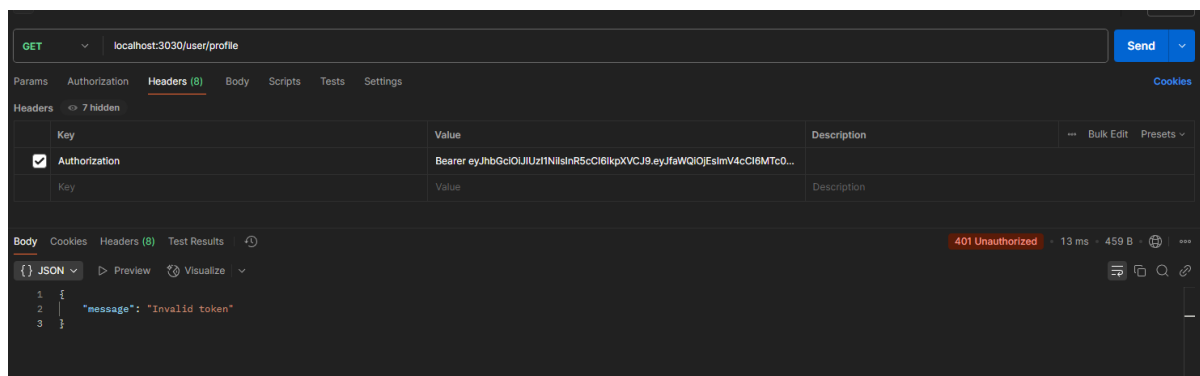
A felhasználó az Authorization fejléccel egy érvényes tokennel hitelesítette magát. A szerver 200-OK státuszkóddal reagált.



33. ábra: Tesztelés Postman segítségével, Normál teszt eset

### 2.5.3. Hibás token esetén:

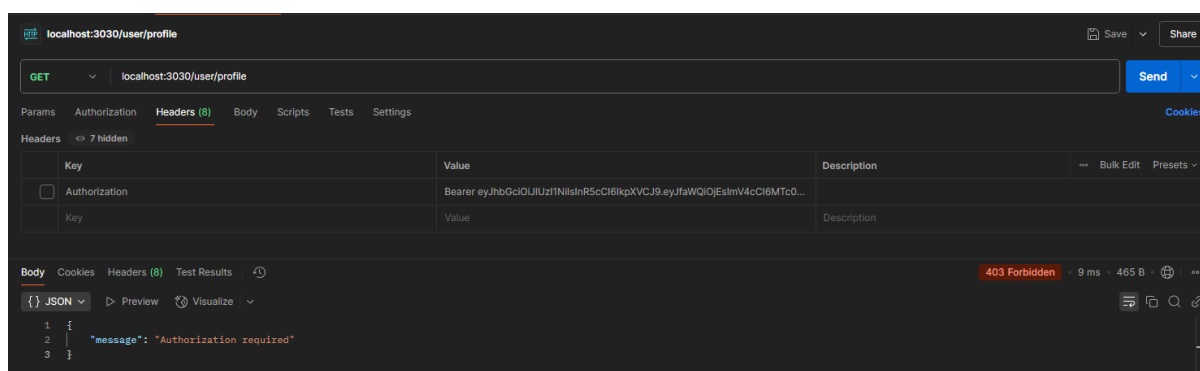
A szerver egy hibaüzenettel tért vissza, „Invalid token”. Ezt a hibát okozhatja egy érvénytelen, rosszul esetleg másik secrettel kiállított vagy lejárt token, normál felhasználás mellett az utóbbi eset valószínűsíthető. Ez esetben a felhasználónak ellenőriznie kell, hogy a bejelentkezése érvényes-e. A szerver 401-Unauthorized státuszkóddal reagált a kérésre.



34. ábra: Tesztet: Érvénytelen token

### 2.5.4. Token nélküli kérés esetén:

A szerver hibaüzenettel tért vissza, „Authorization required”. Ezt a hibát hiányos vagy teljes mértékben hiányzó „authorizaton” fejléc okozhatja. Ez esetben a kérés rosszul van elkészítve, a felhasználónak újra kell próbálkoznia, majd fel kell vennie egy fejlesztővel a kapcsolatot. A szerver 403-Forbidden státuszkóddal reagált a kérésre, azaz a rendszer elutasította a kérést.



35. ábra: Hiányos fejléc

### 2.5.5. Egyéb információ:

A program a különböző végpontok fejlesztése közben folyamatosan tesztelve volt, hogy biztosítsuk a stabil és biztonságos működést. A tesztelés manuálisan és felhasználói visszajelzések alapján történt, így a rendszer valós használati környezetben is ellenőrzésre

került. A fejlesztés során különböző hibaforgatókönyveket szimuláltunk, beleértve az érvénytelen vagy lejárt tokenekkel történő kéréseket, valamint a hiányos hitelesítési fejléceket. A tesztek célja az volt, hogy feltárjuk a potenciális problémákat, és biztosítsuk, hogy a szerver megfelelő státuszkódokkal és egyértelmű hibaüzenetekkel reagáljon. Mivel automatikus tesztek nem készültek, a rendszer működésének ellenőrzése elsősorban manuális teszteléssel és felhasználói visszajelzések kiértékelésével történt. Ez lehetővé tette, hogy az esetleges hibákra gyorsan reagáljunk és javításokat végezzünk a fejlesztési folyamat során.

## **2.6. Továbbfejlesztési lehetőségek**

### **2.6.1. Beszédalapú interakció:**

A karakterekkel való szóbeli beszélgetés bevezetése, ahol a mesterséges intelligencia hangalapú válaszokat is ad valós időben, választható hangon, különböző gyorsaságú beszédstílusban.

Ez a fejlesztés hasznos lehet akár az egészen kiskorú gyermekek számára, akik még nehézkesen használják a billentyűzetet, vagy épp a szépkorú, időskorú emberek számára, akik fitten szeretnék tartani tudásukat, de nehezen használják a digitális eszközöket.

### **2.6.2. Pontgyűjtési rendszer:**

Jutalmak beépítése a tanulási folyamat ösztönzésére, „streak” rendszer, akár jutalmak kiváltása az aktivitással megszerzett kreditből. Egyedi tartalmak, privát tanórák válnának elérhetővé a minket támogató, vagy az elég szorgalmas diákoknak. Ezek a konzultációk különböző közösségi platformjaink egyikén, például Discordon lehetnének megtartva.

### **2.6.3. Mobilalkalmazás:**

Egy könnyebben használható, reszponzívabb, kényelmesebb mobil megoldás, akár offline tartalmak elérése érdekében, vagy a régebbi csevegési előzmények olvasása, tanulmányozása érdekében. Akár egy kisebb lokális nyelvi modell beépítése, amit offline is használhat a felhasználó.

### **2.6.4. Személyre szabott ajánlások:**

Az AI vagy egy erre kifejlesztett algoritmus figyeli a felhasználók érdeklődési körét és tanulási szokásait, majd személyre szabott javaslatokat ad, akár emailben, vagy mobil alkalmazás esetében „pop-up” értesítésekkel üzen a felhasználónak.

Kisebb felmérések, ahol a rendszerünk megállapíthatja, hogy milyen területen kell még fejlesztenie magát a diáknak.

### **2.6.5. Mesterséges intelligencia fejlesztése:**

Finomhangolt modell készítése egyedi betanított adatokkal, hogy a válaszok hitelesebbek és pontosabbak legyenek.

Egy teljes mértékben magyarra hangolt saját nyelvi modell, amely még jobban teljesít nyelvtani feladatokban. Azonban ennek jelenleg jelentős a költség, hardver és időigénye.

### **2.6.6. Jobb adatbiztonság:**

Kétfaktoros hitelesítés, titkosított adatátvitel és biztonságos adatkezelés további fejlesztése.

Habár sok kérdésben, például regisztráció, bejelentkezés vagy a jelszót visszaállító oldalaink háttérében titkosítva küldjük az adatokat, még nem teljesen tökéletes az alkalmazás, habár a globális jogszabályoknak jelenleg is tökéletesen megfelel és sok hazai platformot határozottan túlszárnyal.

### **2.6.7. Partnerségek iskolákkal és egyetemekkel:**

Az alkalmazás hivatalos oktatási programba való beillesztése, közreműködés az e-kréta rendszerével a könnyű integrálás érdekében.

Intézmény szintű csoportok létrehozása, a felhasználó regisztrációkor, vagy a személyes adatainál kiválaszthatja az iskolája nevét, megadhatja osztályát, így könnyen szinkronizálható egy tanmenet a diákok között.

### **2.6.8. Küzdelem a magány ellen**

Rengeteg fiatal kiközösítve, magányosan él, a közvetlen környezetében, iskolájában nem talál magához hasonló érdeklődésű és beállítottságú társakat, éppen ezért sokan az interneten próbálnak barátokat találni. Azonban ez sokszor rendkívül nehéz, vannak, akik a Minervához hasonló alkalmazásokat használva beszélgetnek egy mesterséges intelligencia alapú modellel, így úgy érezhetik, hogy egy valós törődő baráttal beszélgetnek, hisz az AI mindig kedvesen, érdeklődően áll a felhasználóhoz, és felveszik annak stílusát.

Éppen ezért a jövőben szeretnék készíteni egy olyan algoritmust, ami a beszélgetések, érdeklődési körök, zenei stílusok és előadók, életkor, osztály, felhasználó iskolájának körzete alapján tesz egy potenciális barátajánlást mind két félnek. Amennyiben mind két felhasználó elfogadja az ajánlást, felvehetik egymással a kapcsolatot.

Ebben játszik fontos szerepet a Discord platform és az ott található hivatalos EduMInerva közösségünk, mind a mellett, hogy itt felvehetik egymással a kapcsolatot a diákok akár hangcsevegés, akár privát beszélgetés formájában, az itt lévő integrációnk és applikációnk a moderációs feladatok ellátása mellett azt is nyomon tudja követni, hogy a szerverünkön lévő felhasználók milyen zenét hallgatnak, videókat néznek, játékokat játszanak és mennyi időt töltenek ezekkel, amennyiben engedélyezték ezen tevékenységek nyomon követését és



hozzárendelték a megfelelő applikációkat (mint például Spotify, Youtube) a Discord fiókjukhoz.

Köszönhetően annak, hogy már több mint két éve hitelesített aktív fejlesztő vagyok a Discord rendszerében, az applikációim hozzáférhetnek ilyen és ehhez hasonló statisztikai információkhoz a megfelelő adatvédelmi és biztonsági intézkedések betartása jegyében, ezért azt érzem, egy ilyen algoritmus létrehozása egy igazán nehéz, de nem lehetetlen és hiánypótló feladat.

### 3. Összegzés

A munka folyamán számos kihívásnak néztem elébe, a csapatmunka megszervezése is jelentős kihívást jelentett. Az eddigi projektjeim többnyire önállóan vagy kisebb segítségekkel teljesítettem, ám csapatban most dolgoztam először. Hamar problémát szült, hogy a projektet csak nagyjából osztottuk szét, és nem egy logikus felépített gondolatmenet alapján, így előfordult, hogy a feladat egy részét több mint egy hónapig nem bírtuk tesztelni éles környezetben, mert a frontend nagy lemaradásban szenvedett. Ezek után, a munkát konkrét feladatokra és részegységekre bontottunk, például a Fórum teljes backend és frontend logikáját én készítettem, viszont a dizájn megtervezése már nem az én feladatom volt. Ennek a megoldásnak köszönhetően mindenki a maga tempójában tudott haladni.

Mindezek mellett a túlzott maximalista hozzáállásom úgy érzem kissé túlterhelte a csapattársaim. Minden kis apró hibalehetőséget, egy ilyen kis csapattal, egy ilyen nagy projekt mellett nem lehet figyelembe venni, azonban úgy érzem ez többnyire így is sikerült.

Az ötlet megszületésekor nem is gondoltuk, hogy mennyi mindennek utána kell majd járnunk, rengeteg jogi, etikai és technikai kérdés merült fel, amire olykor még tanáraink sem tudták a választ, hiszen a mesterséges intelligencia egy annyira új, és gyorsan fejlődő világ, hogy szinte lehetetlen minden ágával képben lenni, így kénytelen voltam magam utánajárni, és megtanulni értelmezni a hivatalos dokumentációkat és felhasználói feltételeket, ami egy unalmas és monoton feladat, de sajnos kötelező része a fejlesztésnek, de programozóként a jövőben valószínűleg egyre több hasonló feladatban lesz részünk. A gépi tanulás és a mesterséges intelligencia és annak tanítása, megszemélyesítése a magyar jogban és szabályozásban jelenleg egy szürke zóna, de még globálisan sincs pontosan meghatározva bizonyos esetekben.

Ugyanakkor azt érzem, sikerült egy olyan munkát kiadni a kezemből, amire büszke lehetek, és sokat fejlődhettem a program tervezése mellett. Rengeteg új technológiát és funkciót sikerült implementálnom, ami segített egy magasabb szintre lépni a programozás világában. Jobban megismerhettem, hogy hogyan is épül fel egy komplexebb applikáció háttere, a külsős modulok és egyéb szolgáltatások beépítése.

Megismerkedtem rengeteg új modullal, algoritmussal, és közelebb kerültem a mindennapokban használatos alkalmazások működésének megértéséhez, így már jobban átlátom miért mondják egy szoftverre vagy weboldalra, hogy nem megbízható, vagy elavult. Ezen tudások birtokában úgy érzem, sikerült egy olyan platformot létrehozni, ami a jelenlegi

technikának megfelelő mind biztonsági, mind felépítési szempontból, és még sokáig egy működő és korszerű alapot fog biztosítani az ötletünknek.

## 4. Forrásmegjelölés

### 4.1. Dokumentációk:

Bun:

<https://bun.sh/docs>

MDN Web:

<https://developer.mozilla.org/en-US/>

NodeJS:

<https://nodejs.org/docs/latest/api/>

Nodemailer:

<https://www.nodemailer.com/>

React:

<https://react.dev/learn>

Create React App:

<https://create-react-app.dev/docs/deployment/>

### 4.2. Egyéb források:

Példákhoz használt képek:

<https://www.shutterstock.com/>

## 5. Ábrajegyzék

1. ábra: Az oldal elérési címe.....	11
2. ábra: Főoldal, körhinta elem .....	12
3. ábra: Főoldal mobil nézetben, sötétmódban .....	13
4. ábra: Célkitűzések .....	14
5. ábra: Az oldal lábjegyzete.....	14
6. ábra: Rólunk oldal.....	15
7. ábra: Regisztrációs felület.....	16
8. ábra: Regisztrációs hiba .....	17
9. ábra: Regisztrációs email .....	17
10. ábra: Regisztrációt megerősítő oldal.....	18
11. ábra: Bejelentkezés oldal .....	19
12. ábra: Jelszó visszaállító email .....	20
13. ábra: Jelszó visszaállító oldal.....	20
14. ábra: Bejelentkezés új eszközről.....	21
15. ábra: Beszéljess! menüpont.....	22
16. ábra: Beszélgetés kinézete .....	23
17. ábra: Egyenlet levezetés bemutatása.....	24
18. ábra: Fórum oldal.....	25
19. ábra: Discord fórum szinkronizáció.....	26
20. ábra: Válasz üzenet .....	26
21. ábra: Fiókom oldal .....	27
22. ábra: Fiók oldal, felugró ablak .....	28
23. ábra: Újra aktiválás megerősítő email.....	29
24. ábra: 404 az oldal nem található .....	29
25. ábra: Táblák közötti kapcsolat - <a href="https://dbdiagram.io/">https://dbdiagram.io/</a> .....	41
26. ábra: RSA titkosító algoritmus.....	42
27. ábra: Üzenet küldő funkció .....	44
28. ábra: Tuning betöltő modul.....	45
29. ábra: ModelLoader konzoli kimenet - példa.....	46
30. ábra: Hibátlan konzol példa .....	49
31. ábra: Bun keretrendszer hibakezelés példa .....	50
32. ábra: Authentikációs ábra .....	51

33. ábra: Tesztelés Postman segítségével, Normál teszteset.....	52
34. ábra: Teszteset: Érvénytelen token .....	53
35. ábra: Hiányos fejléc .....	53